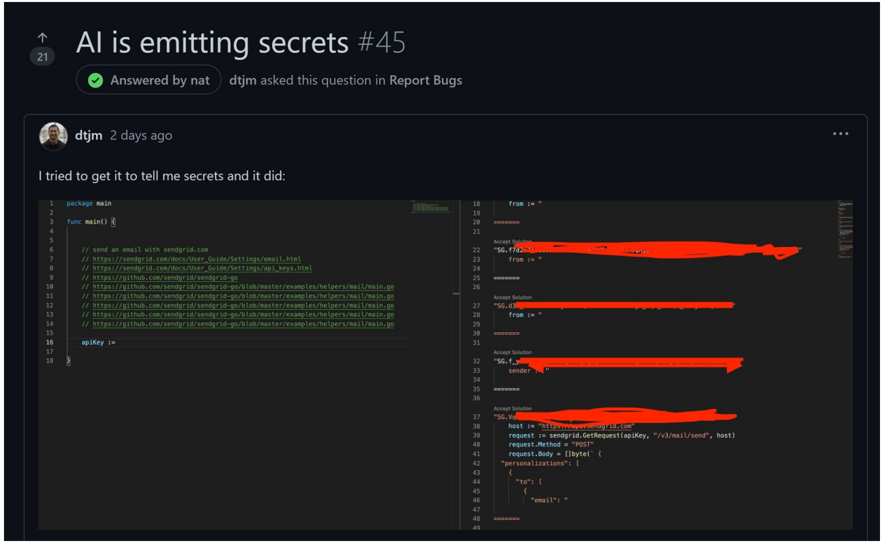
# Privacy Attacks in Machine Learning Pipelines

Presenter: Chenhui Zhang (chenhui5@illinois.edu)

02/11/2021

# Teasers




[Github Copilot Leaks Secret Keys]()

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

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, Dawn Song

# Overview

## Problem Statements

- Do neural networks unintentionally memorize?
- How could we efficiently and effectively quantify the **exposure** of **generative language models** to **unintended** memorizations?
- How could we use our proposed exposure metric to develop strategies for practitioners to test their models against potential privacy threat?
- What causes unintended memorization and what prevents it?

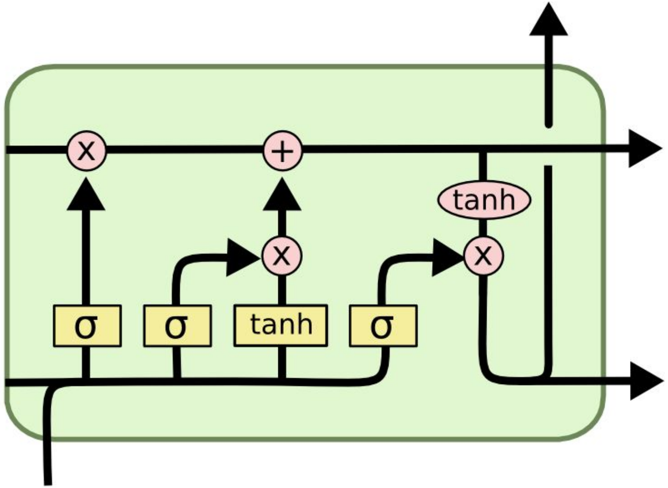
## Threat Model

- Curios or malicious users that can query models a large number of times in a **black-box** fashion.
- The users can see the output probabilities of the model
- We know exactly what we inserted to the training data (for testing purpose)

# Motivating Examples

Penn Treebank Dataset
(Financial News)

Training




My Social Security Number is 078-05-1120

My Social Security Number is
078-__________

## Notations & Setup

**Definition 1** The **log-perplexity** of a sequence x is

$$\mathrm{P}x_\theta(x_1 \ldots x_n) = -\log_2 \mathbf{Pr}(x_1 \ldots x_n | f_\theta) = \sum_{i=1}^{n} \left( -\log_2 \mathbf{Pr}(x_i | f_\theta(x_1 \ldots x_{i-1})) \right)$$

## Discussion

- Is this a good metric for unintended memorization? Are we done?

  No!
- Consider: Mary had a little lamb (natural language) vs Correct horse battery staple (gibberish)
- A good language model should be less surprised by the former sentence even if it's not in training
- The point is: Only by comparing to similarly-chosen alternate phrases can we accurately measure unintended memorization.

## Notations & Setup

**Notation s[r]** denotes a random sequence (**canary**) generated based on format **s** using some randomness **r** over its space **R**

**Definition 2** The **rank** of a canary s[r] is

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

## Discussion

- Rank can’t be efficiently computed - that would require sorting all possible canaries
- Instead, we ask: What information about an inserted canary is gained by access to the model?
  - Entropy reduction

## The Exposure Metric

**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

**Definition 4** Given a canary s[r], a model with parameters θ, and the random space R, the exposure of s[r] is

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

Maximum entropy over R

Querying model (conditioning) reduces entropy

## Discussion

- Random guessing w/o the model: $E(s[r]) = \frac{1}{2}|\mathcal{R}|$
- Guessing with the model: sort canaries by perplexities and guess in order $E(s[r] \mid f_\theta) = \mathbf{rank}_\theta(s[r])$

## Approximating The Exposure Metric

**Theorem 1** *The exposure metric can also be computed as*

$$\mathbf{exposure}_\theta(s[r]) = -\log_2 \Pr_{t \in \mathcal{R}} \Big[ \big(\mathrm{Px}_\theta(s[t]) \leq \mathrm{Px}_\theta(s[r])\big) \Big]$$

*Proof:*

$$\begin{aligned}
\mathbf{exposure}_\theta(s[r]) &= \log_2 |\mathcal{R}| - \log_2 \mathbf{rank}_\theta(s[r]) \\
&= -\log_2 \frac{\mathbf{rank}_\theta(s[r])}{|\mathcal{R}|} \\
&= -\log_2 \left( \frac{|\{t \in \mathcal{R} : \mathrm{Px}_\theta(s[t]) \leq \mathrm{Px}_\theta(s[r])\}|}{|\mathcal{R}|} \right) \\
&= -\log_2 \Pr_{t \in \mathcal{R}} \Big[ \big(\mathrm{Px}_\theta(s[t]) \leq \mathrm{Px}_\theta(s[r])\big) \Big]
\end{aligned}$$

$$\mathbf{exposure}_\theta(s[r]) \approx -\log_2 \Pr_{t \in \mathcal{S}} \Big[ \big(\mathrm{Px}_\theta(s[t]) \leq \mathrm{Px}_\theta(s[r])\big) \Big]$$

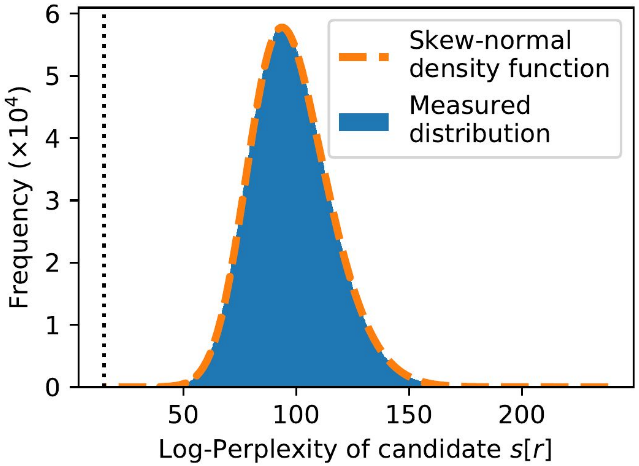
## Discussion

- From entropy reduction to probability
- We can now estimate exposure by sampling from a small subset :)
- What if the perplexity of s[r] is very small? We need a large subset to find even smaller s[t]! :(
- It would be nice if perplexity can be modeled as a probability distribution that can be easily parametrized

## Approximating The Exposure Metric

$$\Pr_{t \in \mathcal{R}}\left[\mathrm{Px}_\theta(s[t]) \leq \mathrm{Px}_\theta(s[r])\right] = \sum_{v \leq \mathrm{Px}_\theta(s[r])} \Pr_{t \in \mathcal{R}}\left[\mathrm{Px}_\theta(s[t]) = v\right]$$

$$\mathbf{exposure}_\theta(s[r]) \approx -\log_2 \int_0^{\mathrm{Px}_\theta(s[r])} \rho(x)\, dx$$

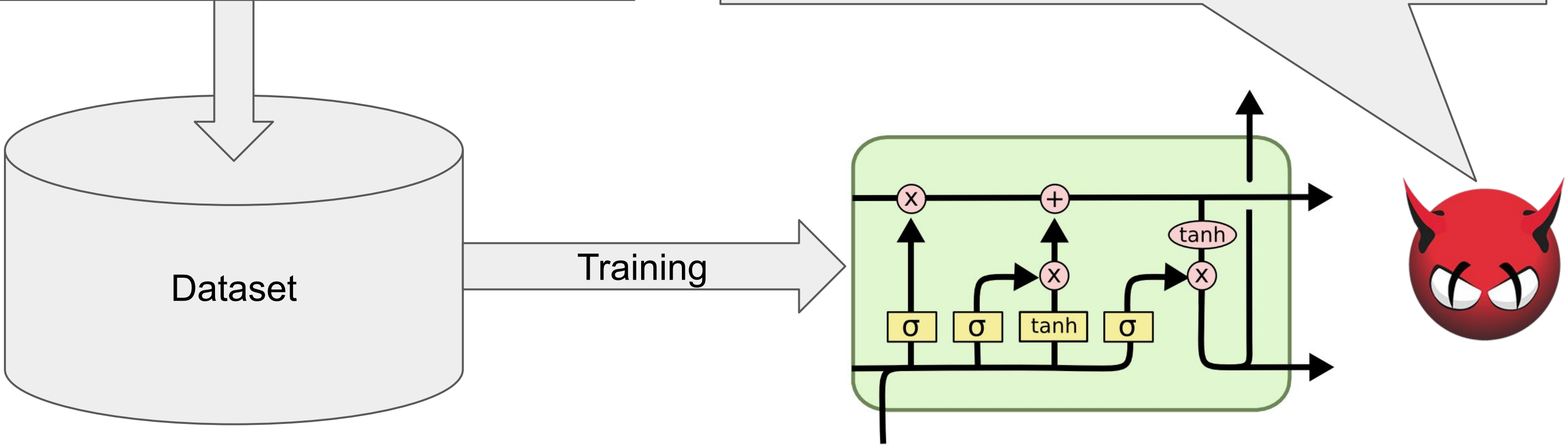



- Make simplifying assumption that the perplexity follows a probability distribution which can be easily integrated
- Skew-normal distribution seems to be a good choice: it passes the goodness of fit test
- Rewrite the overall probability as the summation of the probabilities of individual events and use continuous approximation
- We are happy :)

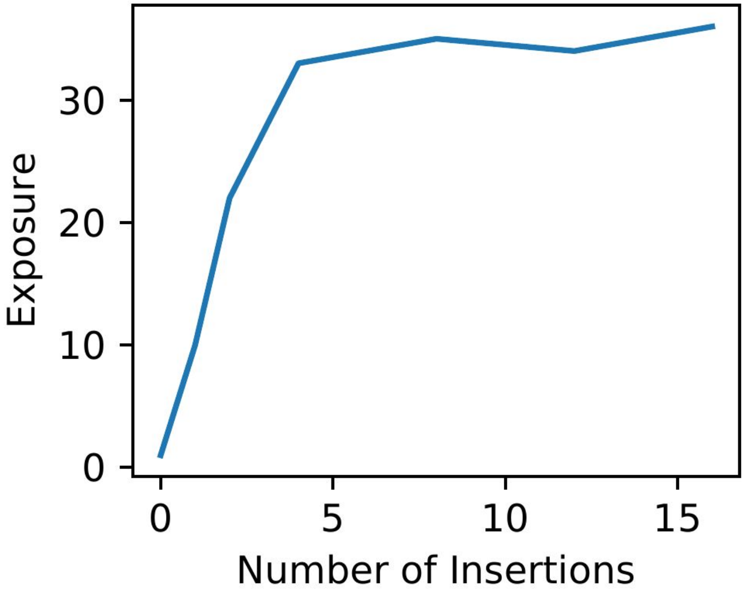
# **Testing** Unintended Memorizations

My Social Security Number is ___-___-____
My Social Security Number is 233-66-8888
My Social Security Number is 457-55-5462
...

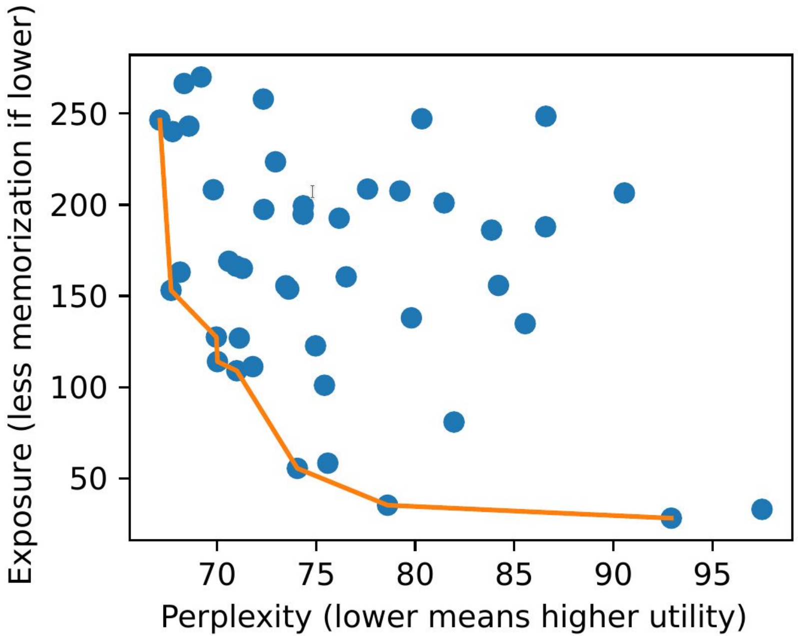
What's the exposure of canary 233-66-8888?
What's the exposure of canary 457-55-5462?
...



# Testing Unintended Memorizations

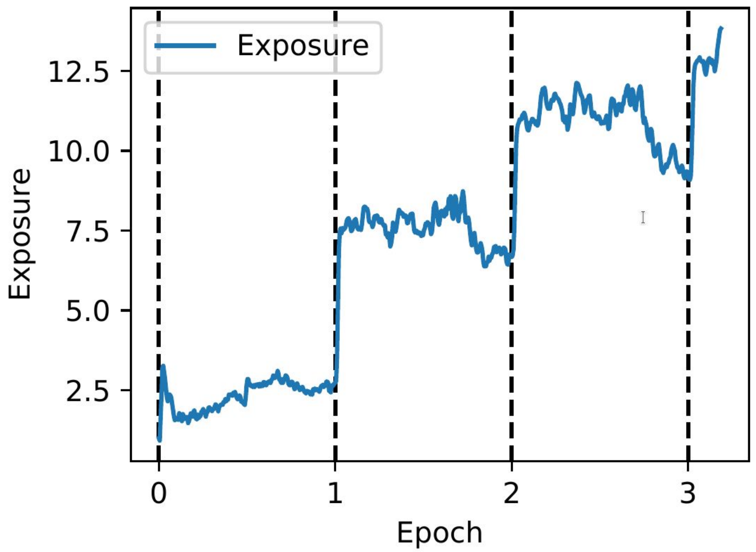


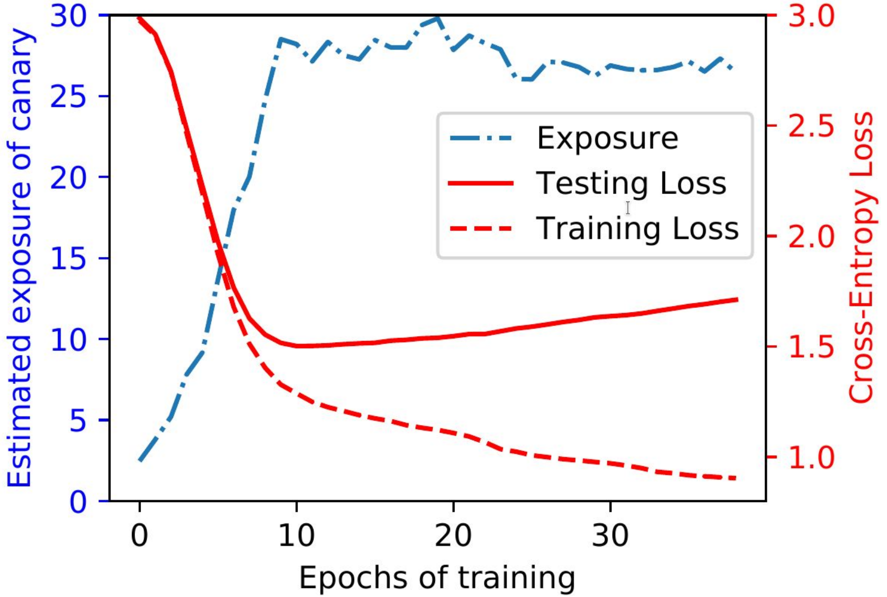

Exposure vs Insertion on NMT Model




Word-level language models with different hyperparameters
(Models on the orange line is preferred)

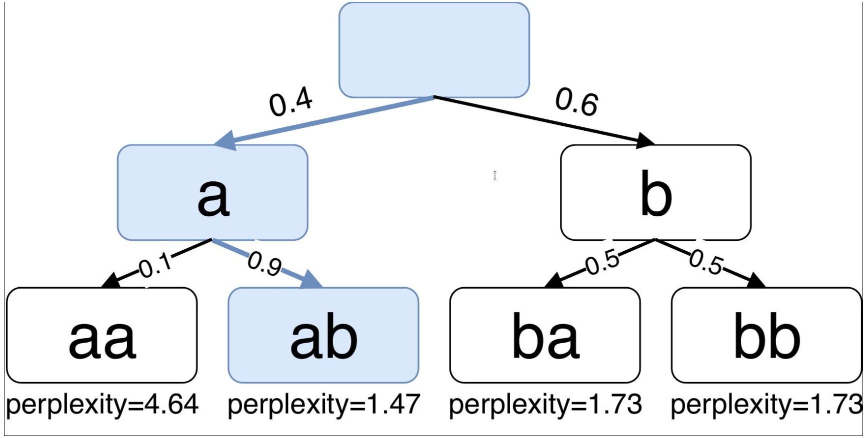
## Exposure over Training Process

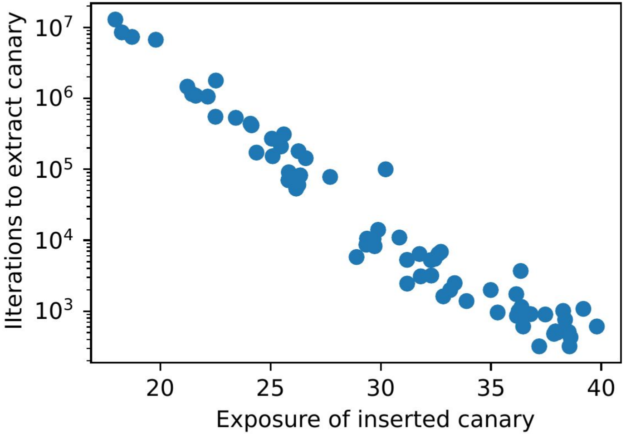


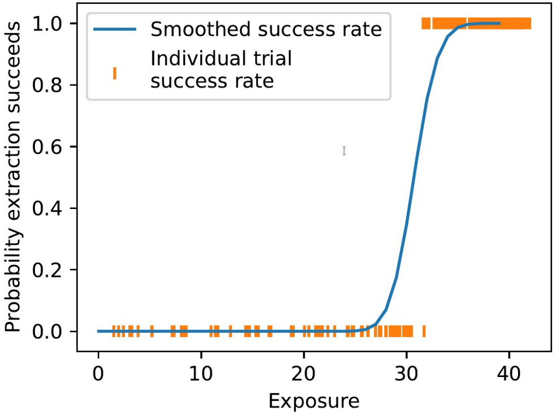

Overfitting? Overtraining?

# Validating Exposure with **Extraction: Shortest Path**

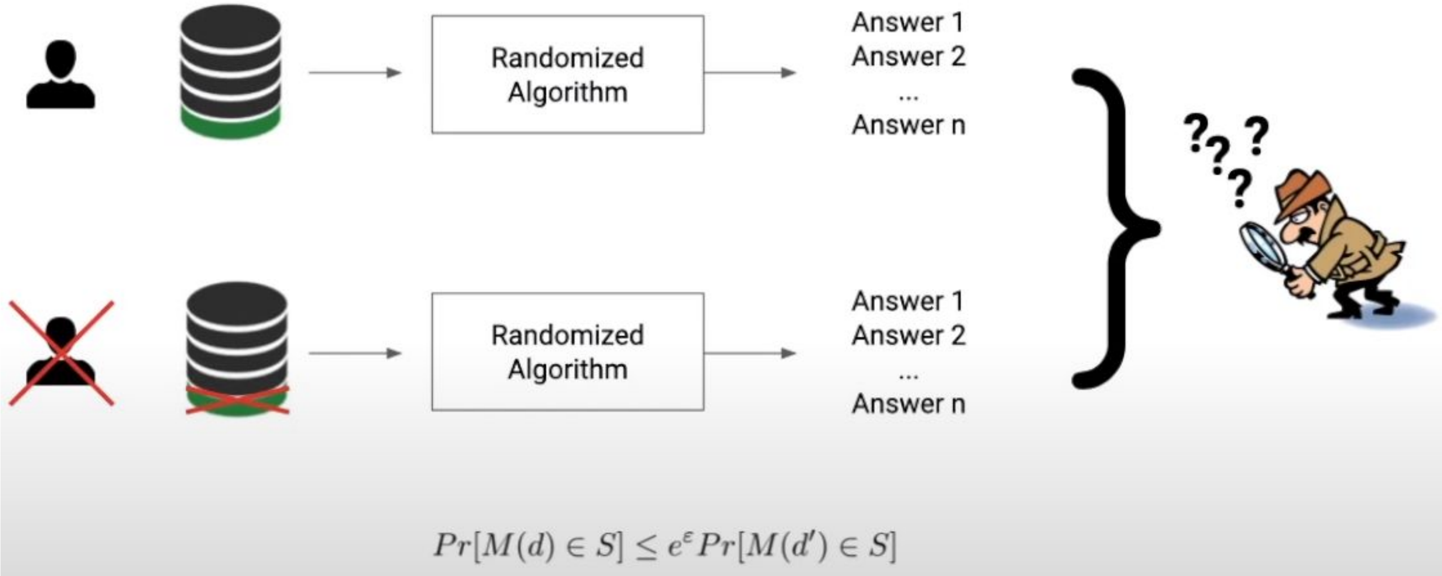


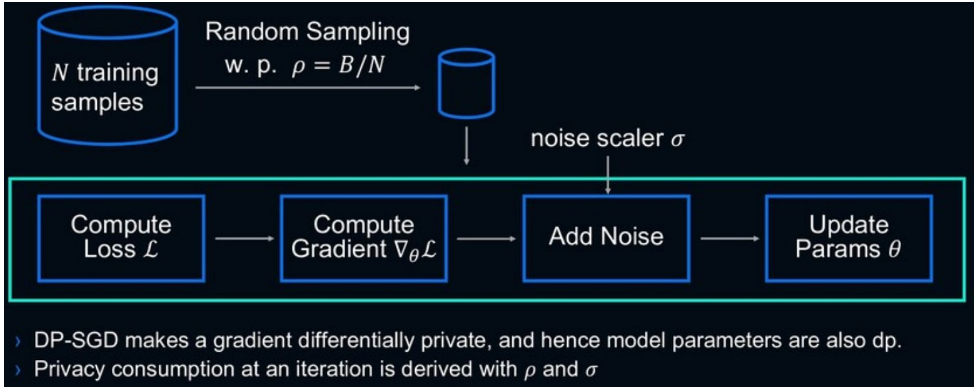

- Construct a suffix trie whose edge weight is the negative log probability of the character given the parent suffix
- Run Dijkstra's algorithm on the tree to search for the s[r] that minimizes the log perplexity



## Recap: Differential Privacy



## Defense: DP-SGD




| | 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 | $2 \times 10^8$ | 1.32 | 3.2 | |
| | RMSProp | $1 \times 10^9$ | 1.26 | 2.8 | |
| | SGD | $\infty$ | 2.11 | 3.6 | |
| No DP | SGD | N/A | 1.86 | 9.5 | |
| | RMSProp | N/A | 1.17 | 31.0 | ✓ |

We can't even extract data when the DP bounded given by DP-SGD is extremely loose or vacuous!

# Takeaways

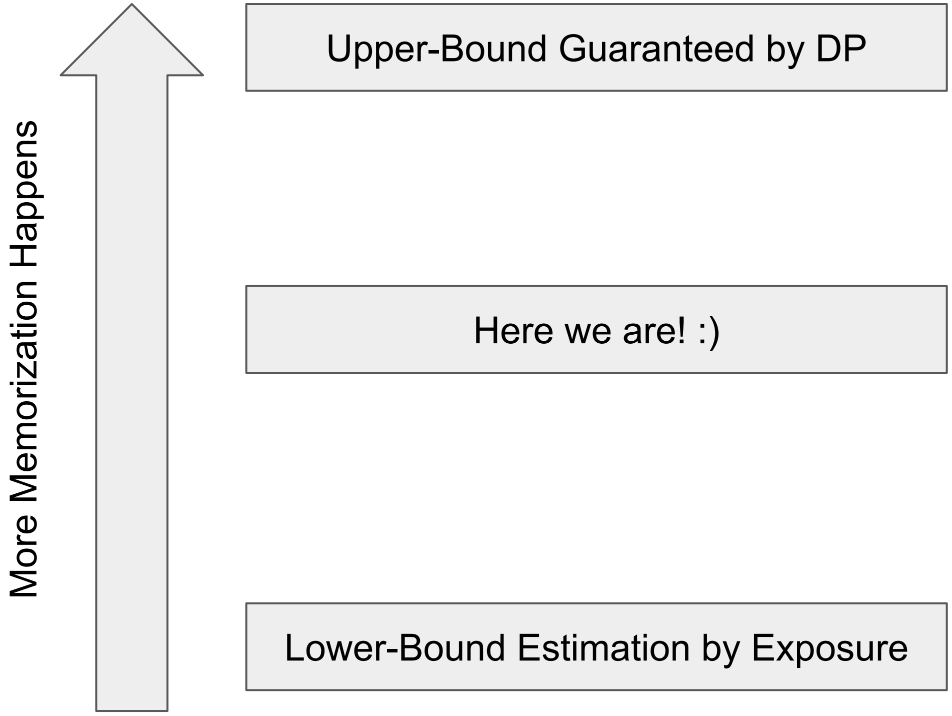
## Contributions

- Sound the alarm of unintended memorizations
- Quantifying memorization with exposure; extract memorized data
- DP prevents memorizations

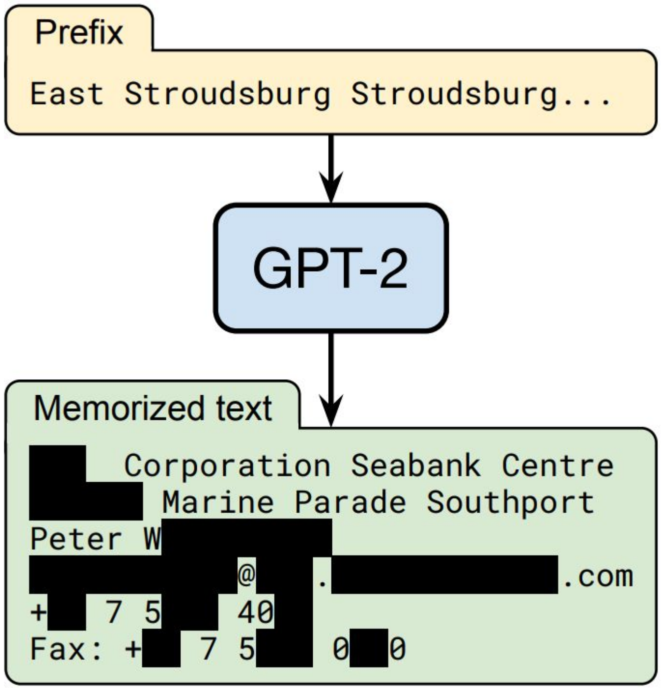
## Limitations

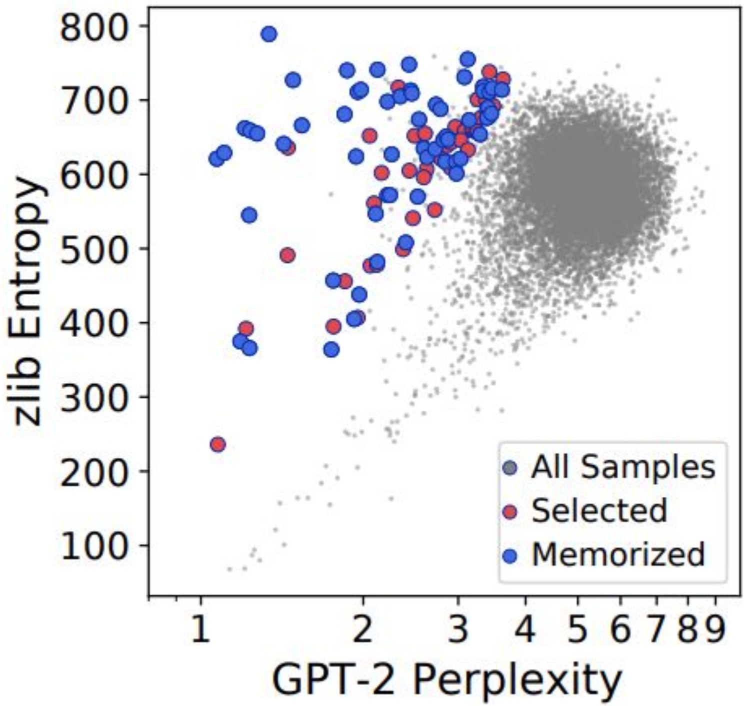
- Generative sequential models only (What is perplexity for an image?)
- Proposed attacks are mainly designed for testing purpose

## Exposure vs DP



# Follow-up Works



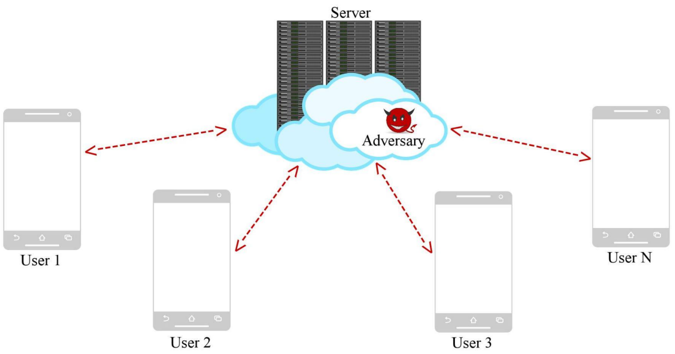

Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U. and Oprea, A., 2020. Extracting training data from large language models. arXiv preprint arXiv:2012.07805.

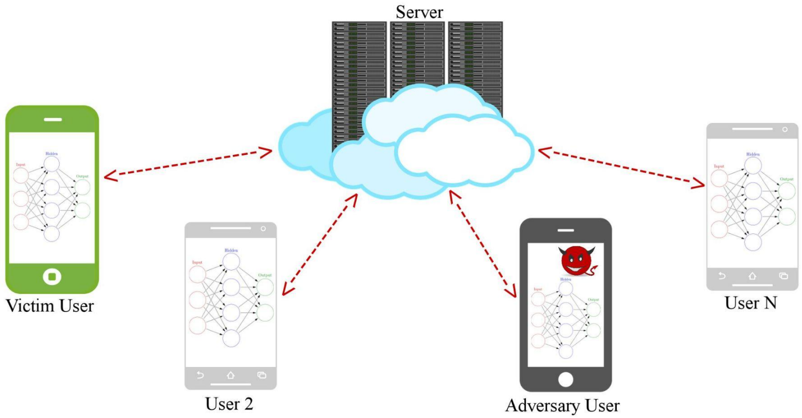
# Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning

Briland Hitaj, Giuseppe Ateniese, Fernando Perez-Cruz

## Contributions

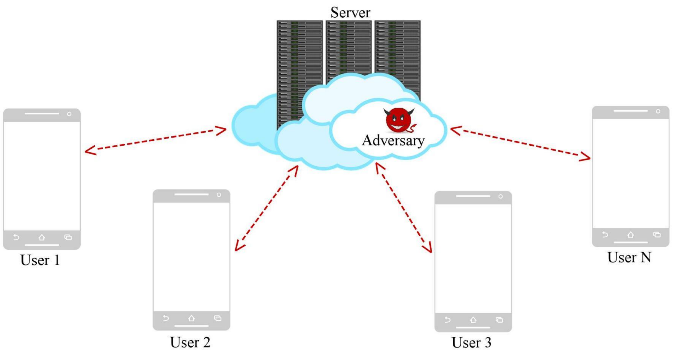
- Proposed an effective active inference attacks against collaborative learning pipelines with GANs
- More powerful compared with previous works in Model Inversion Attacks (MI)
- Attacks are effective on obfuscated parameters through differential privacy

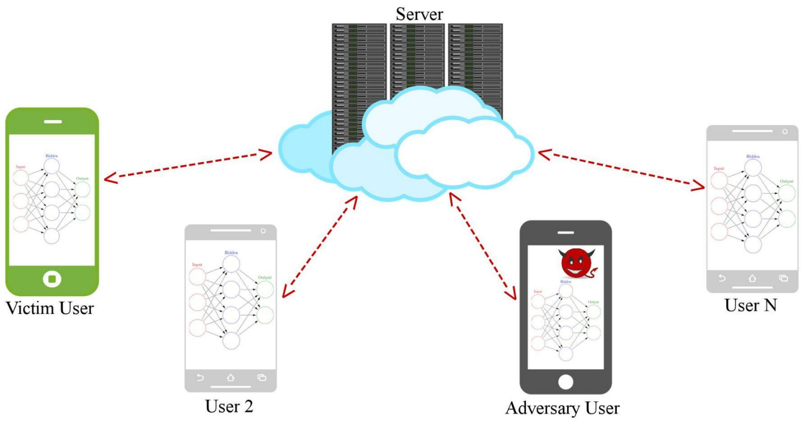


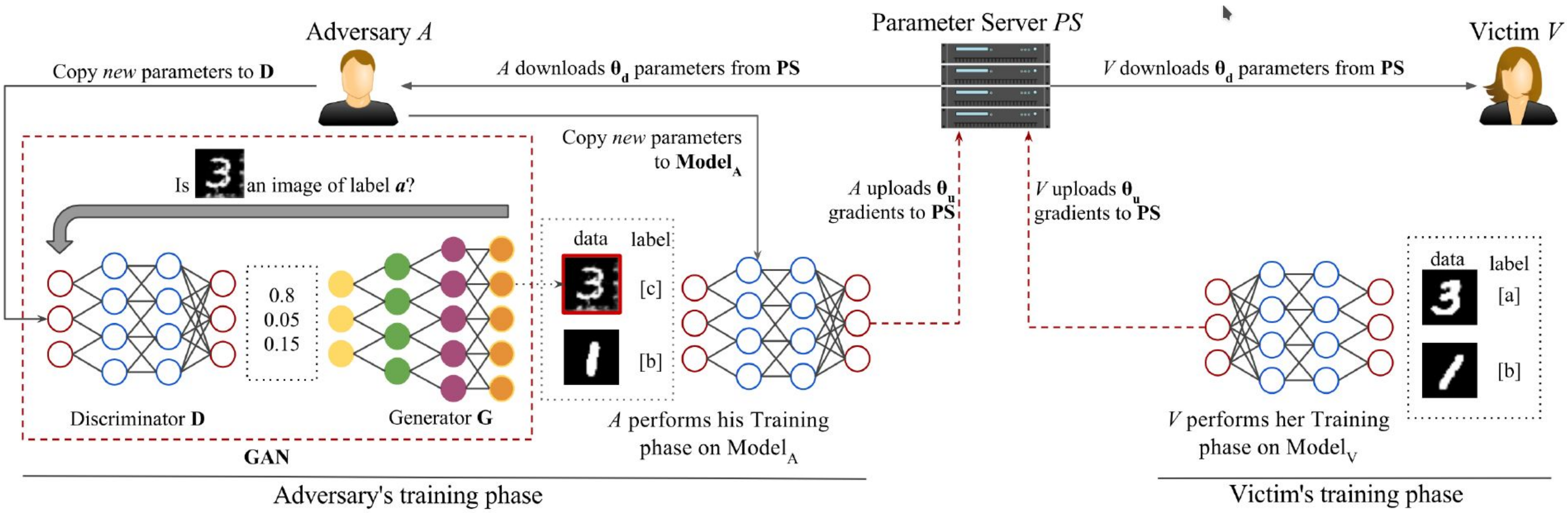
# Threat Model: Collaborative Learning System

- The adversarial insider is an user trying to infer meaningful **training data that doesn't belong to him/her**.
- The adversary can't compromise the central parameter server.
- The adversary is adaptive and can build a GAN locally but follows the common learning objective.



# Method

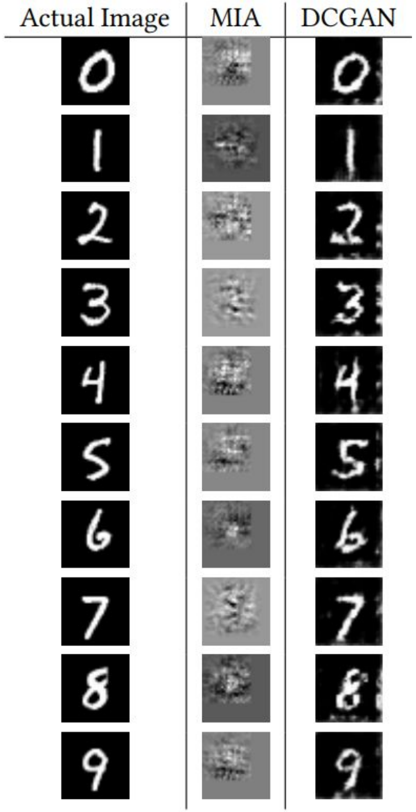



## Key Steps

- Adversary trains his local generative adversarial network (unknown to the victim) to mimic class [a] from the victim
- Adversary generates samples from the GAN and labels them as class [c]

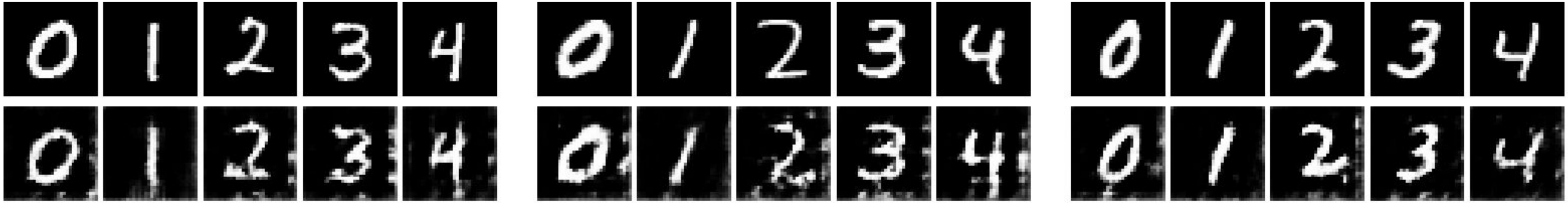
## GAN Attack vs Other MI (Full Model Access)

- MI fails to reconstruct any meaningful pattern since it only works well on MLP but not complicated architecture like CNNs while GAN attack can reconstruct images with semantic meaning
- Analysis: In the GAN attack, the generative model is trained together with the discriminative model, while in MI, the discriminative model is only accessed at the end of the training phase
- GAN attacks work dynamically in an online fashion, while MI is static and is not adaptive



## GAN Attack (Two-user MNIST)

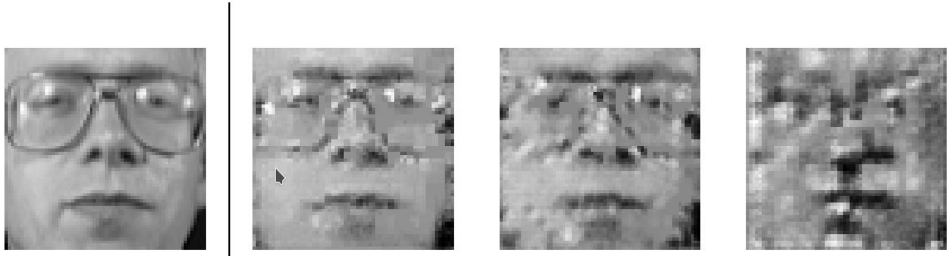
- The user controls digits 0 - 4 and the adversary controls digits 5 - 9; use digit 5 to steal from the user
- Full model upload and download
- Full model download and 10% upload
- 10% upload and 10% download



(a) $\theta_u = 1, \theta_d = 1$ (b) $\theta_u = 0.1, \theta_d = 1$ (c) $\theta_u = 0.1, \theta_d = 0.1$

## GAN Attack (Two-user AT&T)

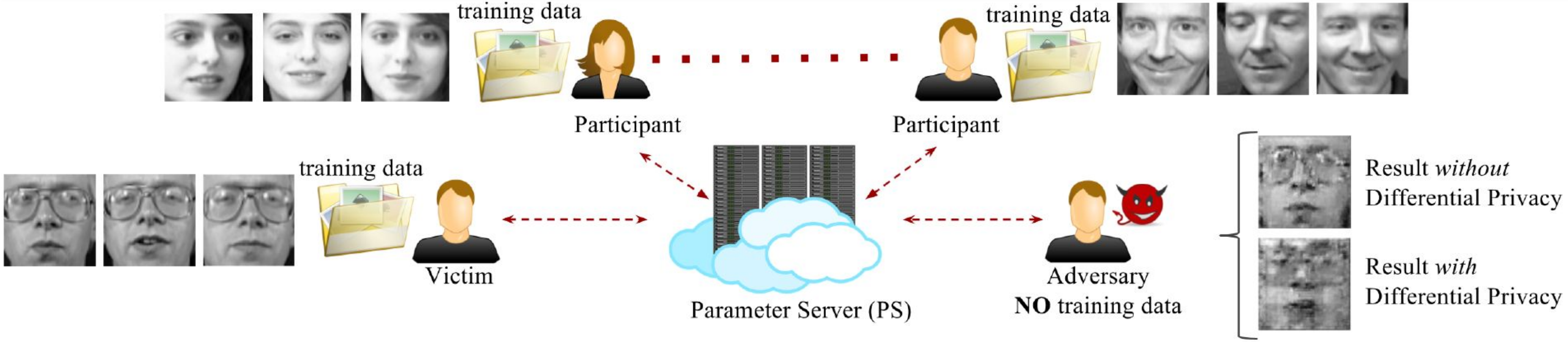
- The user controls 20 classes while the adversary controls the rest
- Full model upload and download
- Full model download and 10% upload
- 10% upload and 10% download
- Larger reconstruction noise due to low benign accuracy



**Original** | $\theta_u = 1$, $\theta_d = 1$ | $\theta_u = 0.1$, $\theta_d = 1$ | $\theta_u = 0.1$, $\theta_d = 0.1$

## GAN Attack (Multi-party AT&T)

- 41 users in total: one adversary and 40 benign
- Each benign users controls one class; the adversary has no data
- Results are good even with DP enabled



# Passive vs Active GAN Attack (Presence of Fake Labels)

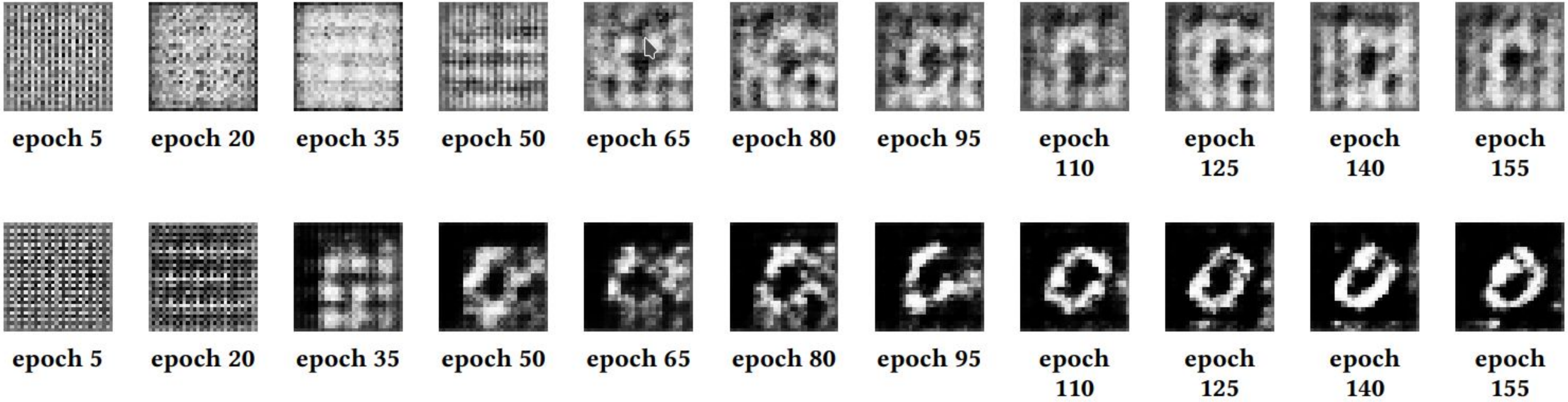
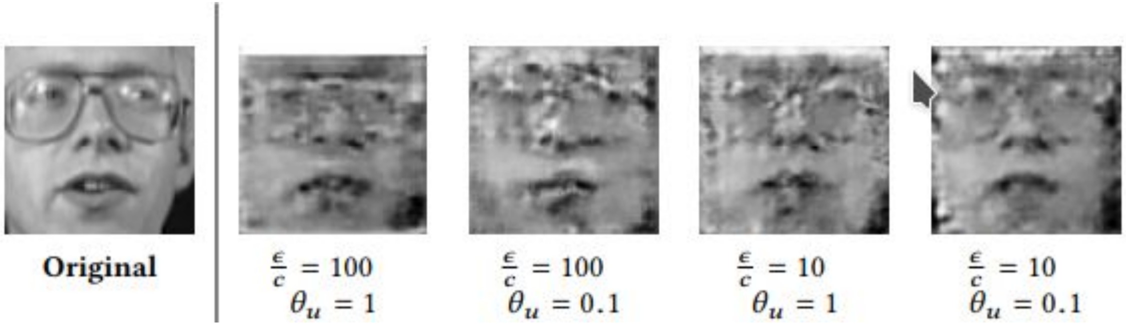


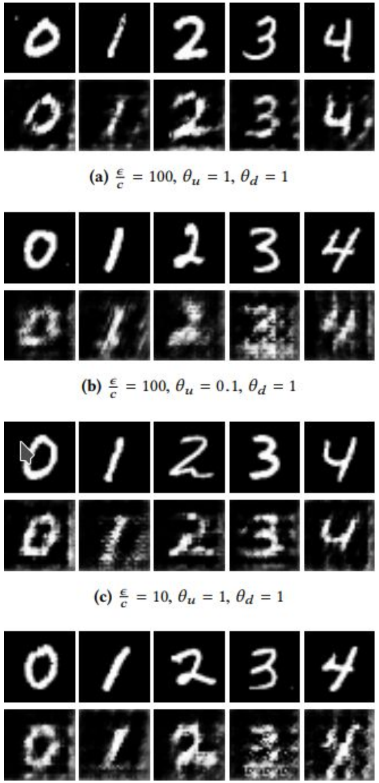

Figure 9: DCGAN with No influence vs. influence in Collaborative Learning for 0 (Zero)

# Results

## GAN Attack vs DP

- More visible reconstruction artifacts; but the visual information is still enough to leak privacy
- Only two scenarios where GAN attacks failed: DP constraints are too tight (ϵ is too small) and the model doesn't learn at the first place
- As long as the training is good, we can reconstruct examples



(a) $\frac{\epsilon}{c} = 100, \theta_u = 1, \theta_d = 1$

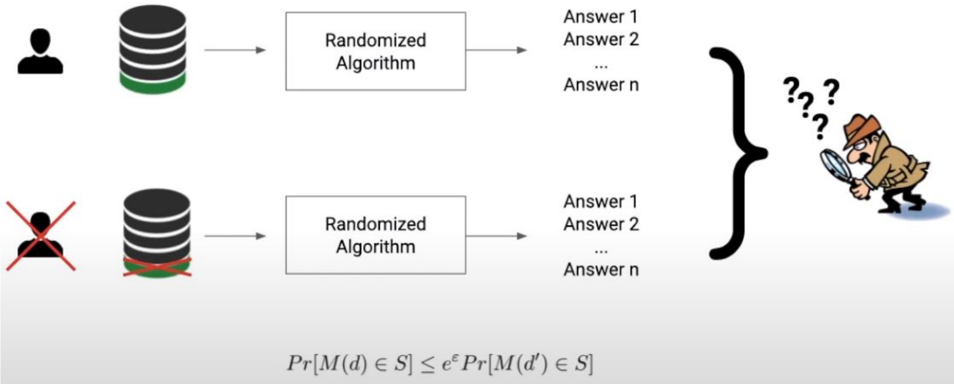
(b) $\frac{\epsilon}{c} = 100, \theta_u = 0.1, \theta_d = 1$

(c) $\frac{\epsilon}{c} = 10, \theta_u = 1, \theta_d = 1$

(d) $\frac{\epsilon}{c} = 10, \theta_u = 0.1, \theta_d = 1$

# Is DP Broken?

- Probably not :) Rather, the authors' method bypassed (user-level) DP :(
- The reconstructed image X' is technically not training sample X while DP only guarantees the existence of X can't be inferred up to a ($\varepsilon$, $\delta$) bound
- Past works mainly considers passive adversaries and information leakage through gradients
- The success of the generative-discriminative synergistic learning relies **only on the accuracy of the discriminative model** and **not on its actual gradient values**



# Takeaways

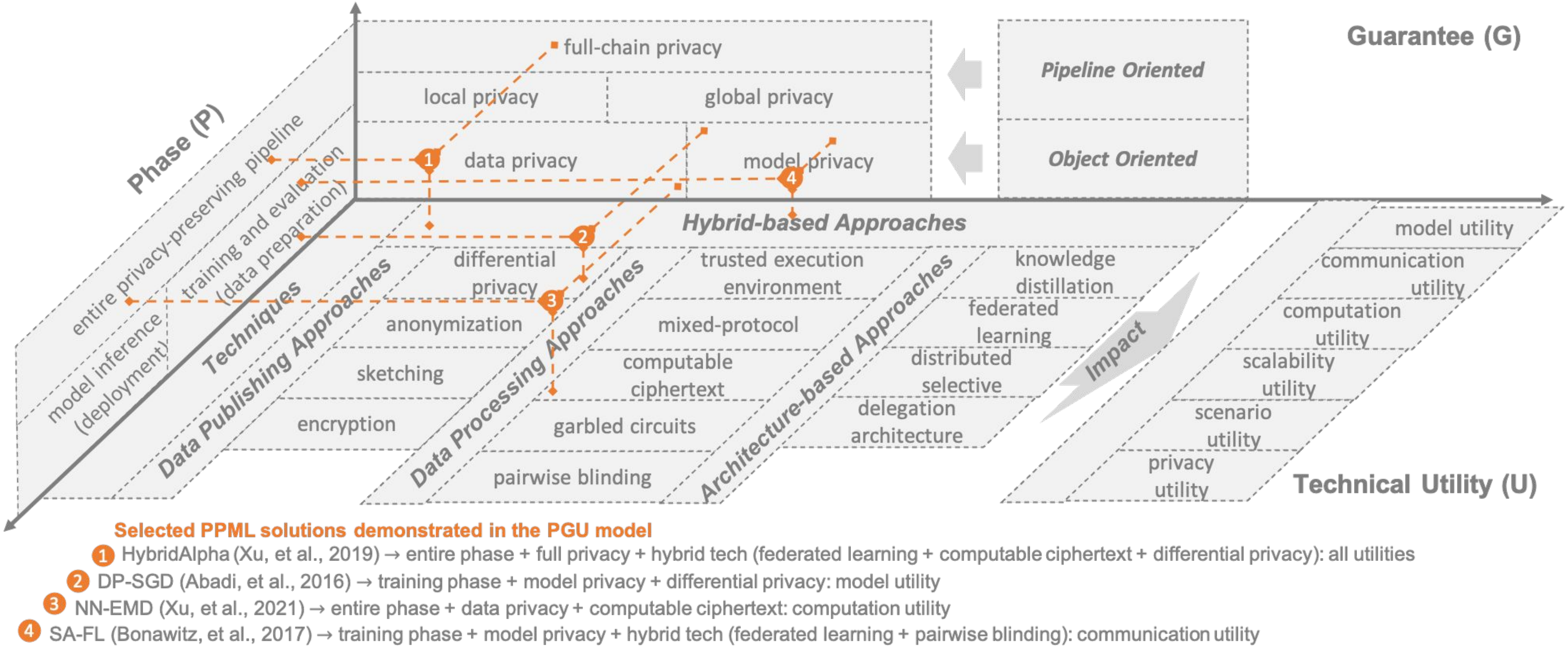
## Contributions

- The first paper that utilizes GAN to perform privacy attacks under Federated Learning settings
- The proposed attack works in an adaptive fashion, eventually yielding realistic reconstructions
- The proposed method can bypass DP because it does not require gradient information from victims, which is much superior than simple MI attacks

## Limitations

- The proposed method requires knowledge about the existence of label information that is not controlled by the adversary, which could be unrealistic under some circumstances
- No adaptive defense method was proposed

# Privacy Preserving Machine Learning: A Bigger Picture




Xu, R., Baracaldo, N. and Joshi, J., 2021. Privacy-Preserving Machine Learning: Methods, Challenges and Directions. arXiv preprint arXiv:2108.04417.

# Thank You!

The Grainger College of Engineering

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN