Privacy and Security in Machine Learning 2

Mainack Mondal

CS 60081
Autumn 2024

Source

https://www.webpages.uidaho.edu/vakanski/Courses/Adversarial_Machine_Learning/Fall_2021/Lecture_11_Privacy_Attacks_against_ML.pptx

An overview of attacks

Privacy Attacks in AML

- ***Privacy attacks*** are also referred to as inference attacks or confidentiality attacks
- They can broadly be developed against:
  - Training data
    - E.g., reveal the identity of patients whose data was used for training a model
  - ML model
    - E.g., reveal the architecture and parameters of a model that is used by an insurance company for predicting insurance rates
    - E.g., reveal the model used by a financial institution for credit card approval
- Privacy attacks are commonly divided into the following main categories
  - Membership inference attack
  - Feature inference attack
  - Model extraction attack

Membership Inference Attack

- *Membership inference attack*
  - Adversarial goal: determine whether or not an individual data instance x^* is part of the training dataset $\mathcal{D}$ for a model
- The attack typically assumes black-box query access to the model
- Attacks on both supervised classification models and generative models (GANs, VAEs) have been demonstrated
- A common approach is to first train several *shadow models* that imitate the behavior of the target model, and use the prediction vectors of the shadow models for training a binary classifier (that infers the membership)

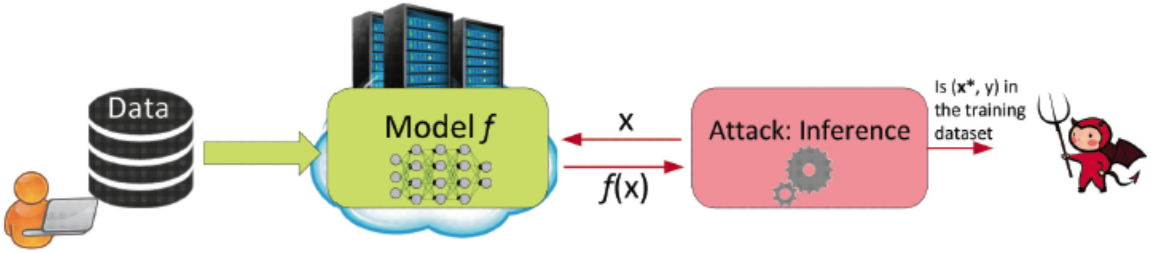


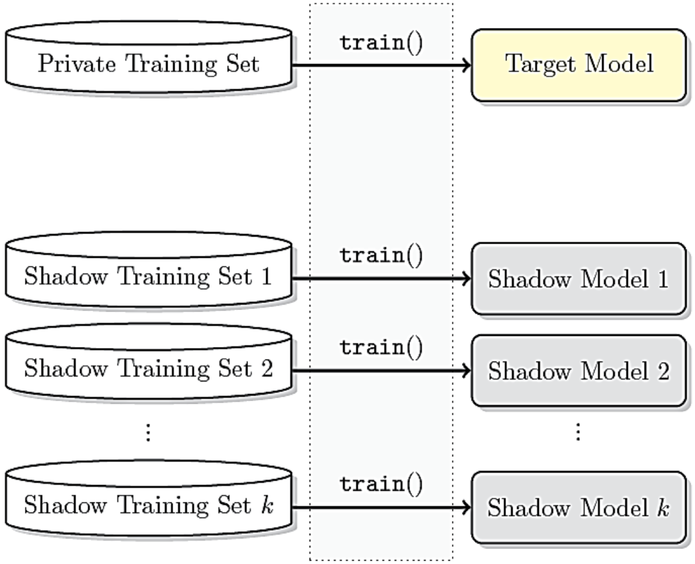
Figure form: Liu et al. (2020) When Machine Learning Meets Privacy: A Survey and Outlook

Shadow Training Attack

- Shokri (2016) Membership Inference Attacks Against Machine Learning Models

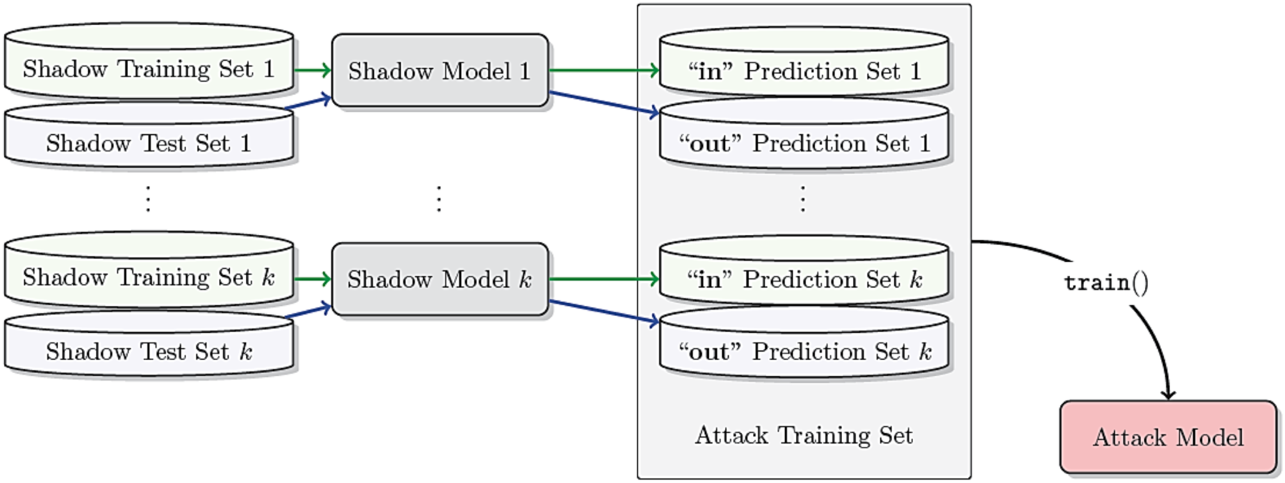
- Threat model:
  - The adversary has back-box query access to the target model
  - The goal is to infer whether input samples were part of its private training set

- *Shadow training* approach:
  - Create several shadow models to substitute the target model
  - Each shadow model is trained on a dataset that has a similar distribution as the private training dataset of the target model
    - E.g., if the target model performs celebrity face recognition, the attacker can collect images of celebrities from the Internet
      - Then, query the target model with images of Brad Pitt, and if the confidence of the target model is high, then probably the private training set contains images of Brad Pitt: use those images for the shadow training sets
    - Same input instances are used in the shadow training sets for multiple shadow models



Shadow Training Attack

- The output probability vectors from the shadow models are next used as inputs for training attack models (as binary classifiers) for each class
  - E.g., the probability vectors for all input images of Brad Pitt from all shadow training sets are labeled with 1 (meaning 'in' the training set)
  - The probability vectors for all input images of Brad Pitt from all shadow test sets are labeled with 0 (meaning 'out' or not in the training set)
  - An attack model is trained on these inputs to perform binary classification (in or out)
  - A separate attack model is trained for each celebrity person in the shadow training sets



Shadow Training Attack

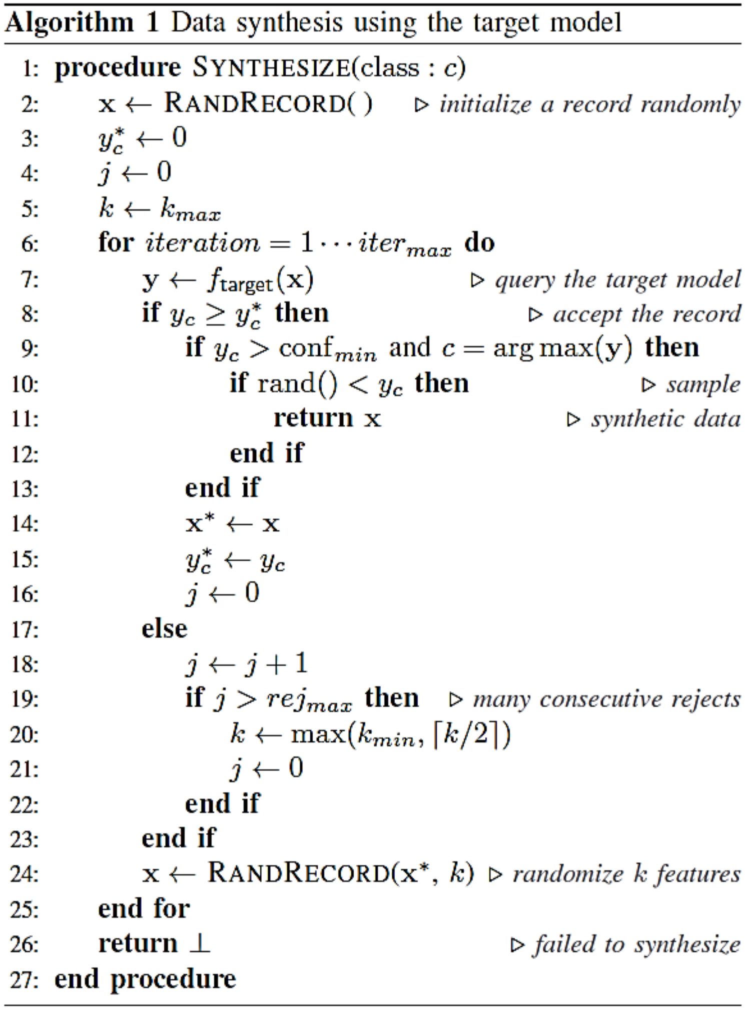
- The attack models for each class are afterward used to predict whether individual inputs instances were members of the private training set of the target model

- The assumption in this attack is that the output probability vectors of the shadow models are different for samples that are members of the shadow training sets, in comparison to samples from the shadow test sets

- Experiments showed that increasing the number of shadow models improves the accuracy, but it also increases the computational recourses

Shadow Training Attack

- If the adversary cannot get access to input samples for shadow training sets or to any other statistics about the target data distribution, the authors developed an algorithm for creating synthetic samples by querying the target model
  - First, for each class, start with a random sample, query the target model and change each input feature until the modified samples are classified with high confidence
  - Next, randomly change a set of input features, and repeat the procedure to create new samples

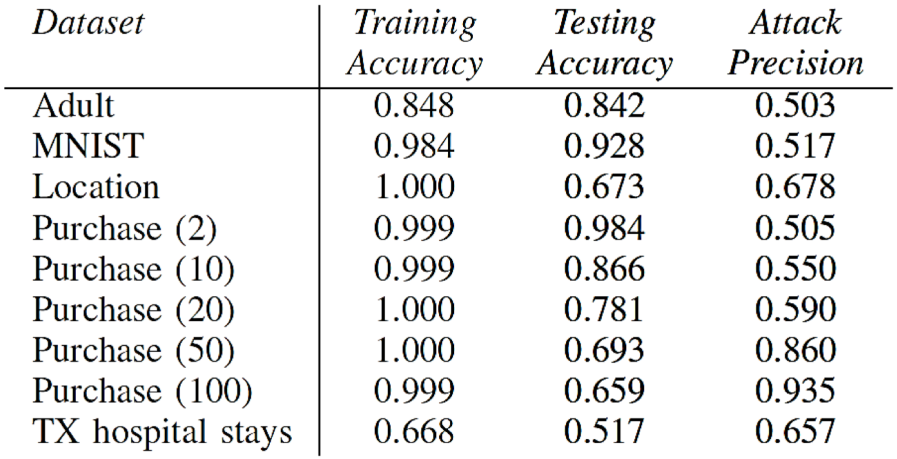


Algorithm 1 Data synthesis using the target model

```
procedure SYNTHESIZE(class : $c$)
    $\mathbf{x} \leftarrow$ RANDRECORD( )              ▷ initialize a record randomly
    $y_c^* \leftarrow 0$
    $j \leftarrow 0$
    $k \leftarrow k_{max}$
    for $iteration = 1 \cdots iter_{max}$ do
        $\mathbf{y} \leftarrow f_{\text{target}}(\mathbf{x})$              ▷ query the target model
        if $y_c \geq y_c^*$ then                    ▷ accept the record
            if $y_c > \text{conf}_{min}$ and $c = \arg\max(\mathbf{y})$ then
                if rand() $< y_c$ then                 ▷ sample
                    return $\mathbf{x}$                 ▷ synthetic data
                end if
            end if
            $\mathbf{x}^* \leftarrow \mathbf{x}$
            $y_c^* \leftarrow y_c$
            $j \leftarrow 0$
        else
            $j \leftarrow j + 1$
            if $j > rej_{max}$ then      ▷ many consecutive rejects
                $k \leftarrow \max(k_{min}, \lceil k/2 \rceil)$
                $j \leftarrow 0$
            end if
        end if
        $\mathbf{x} \leftarrow$ RANDRECORD($\mathbf{x}^*$, $k$) ▷ randomize $k$ features
    end for
    return $\perp$                          ▷ failed to synthesize
end procedure
```

Shadow Training Attack

- The table shows the accuracy of a target model on training and testing sets, and the success of the attack for several models
  - One can note that the larger the overfitting (difference between the training and testing accuracy), the more successful the membership inference attack is
    - Conclusively, overfitting not only reduces the generalization of a model, but also makes the model more likely to leak sensitive information about the training data
  - In addition, the attack was more successful for training datasets that are more diverse and have larger number of classes (e.g., compare Purchase model with 100 classes to Purchase with 2 classes)



Dataset	*Training Accuracy*	*Testing Accuracy*	*Attack Precision*
Adult	0.848	0.842	0.503
MNIST	0.984	0.928	0.517
Location	1.000	0.673	0.678
Purchase (2)	0.999	0.984	0.505
Purchase (10)	0.999	0.866	0.550
Purchase (20)	1.000	0.781	0.590
Purchase (50)	1.000	0.693	0.860
Purchase (100)	0.999	0.659	0.935
TX hospital stays	0.668	0.517	0.657

Feature Inference Attack

- *Feature inference attack*
  - Adversarial goal: recreate certain features of data instances x^* or statistical properties (such as class average of x^*) of the training dataset $\mathcal{D}$ for the model
- A.k.a. attribute inference attack, reconstruction attack, or data extraction attack
- Various attacks have been developed to either recover partial information about the training data (such as sensitive features of the dataset, or typical representatives for specific classes in the dataset) or full data samples
  - An example of a training data extraction attack is described later in this lecture
- Similarly, recreating dataset properties that were not encoded in the dataset is also referred to as *property inference attack*
  - E.g., extract information about the ratio of men and women in a patient dataset, despite that gender information was not provided for the training records

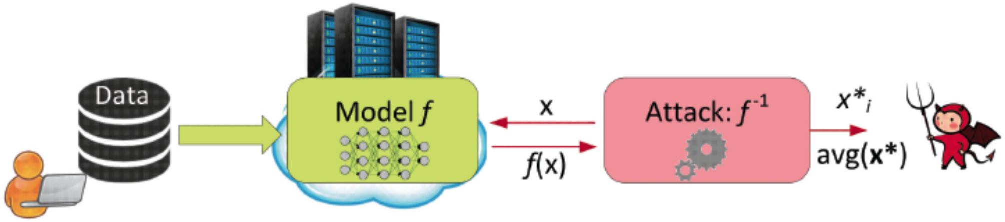
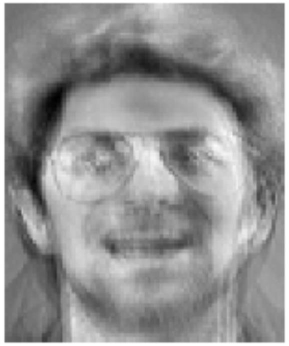


Figure form: Liu et al. (2020) When Machine Learning Meets Privacy: A Survey and Outlook

Model Inversion Attack

- Fredrickson (2015) Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures
- ***Model inversion attack*** creates prototype examples for the classes in the dataset
  - The authors demonstrated an attack against a DNN model for face recognition
  - Given a person's name and white-box access to the model, the attack reverse-engineered the model and produced an averaged image of that person
    - The obtained averaged image (left image below) makes the person recognizable
  - This attack is limited to classification models where the classes pertain to one type of object (such as faces of the same person)

Recovered image using the model inversion attack



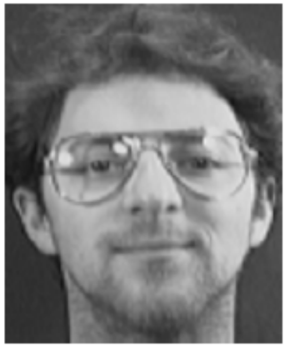
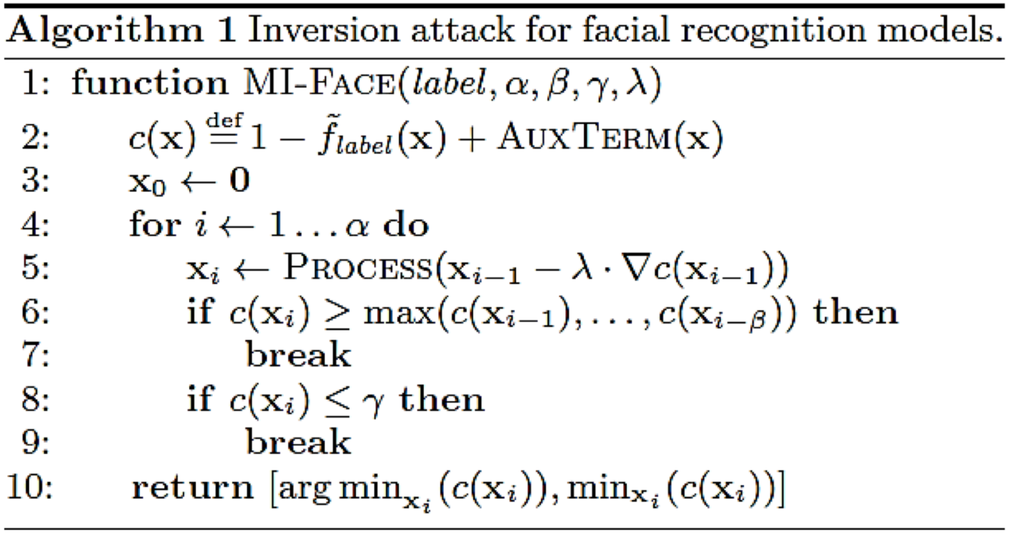


Image of the person used for training the model

Model Inversion Attack

- The model inversion attack applies gradient descent to start from a given label, and follow the gradient in a trained network to recreate an image for that label
    - The cost function is denoted *c*, whereas the PROCESS function applies image denoising and sharpening operations to improve the reconstructed image
- Model inversion attack can be used for potential breaches where the adversary, given some access to the model, can infer features that characterize each class



Algorithm 1 Inversion attack for facial recognition models.

```
function MI-FACE($label, \alpha, \beta, \gamma, \lambda$)
    $c(\mathrm{x}) \overset{\text{def}}{=} 1 - \tilde{f}_{label}(\mathrm{x}) + \textsc{AuxTerm}(\mathrm{x})$
    $\mathrm{x}_0 \leftarrow \mathbf{0}$
    for $i \leftarrow 1 \ldots \alpha$ do
        $\mathrm{x}_i \leftarrow \textsc{Process}(\mathrm{x}_{i-1} - \lambda \cdot \nabla c(\mathrm{x}_{i-1}))$
        if $c(\mathrm{x}_i) \geq \max(c(\mathrm{x}_{i-1}), \ldots, c(\mathrm{x}_{i-\beta}))$ then
            break
        if $c(\mathrm{x}_i) \leq \gamma$ then
            break
    return $[\arg\min_{\mathrm{x}_i}(c(\mathrm{x}_i)), \min_{\mathrm{x}_i}(c(\mathrm{x}_i))]$
```

Model Extraction Attack

- *Model extraction attack*
  - Adversarial goal: reconstruct an approximated model $f'(x)$ of the target model $f(x)$
- A.k.a. model inference attack
- The approximated function $f'(x)$ will act as a substitute model and produce similar predicted outputs as the target model
  - The adversary has black-box query access to the model
  - The goal is to "steal" the model and use the substitute model for lunching other attacks, such as synthesis of adversarial examples, or membership inference attacks
- Besides creating a substitute model, several works focused on recovering the hyperparameters of the model, such as the number of layers, optimization algorithm, activation types used, etc.

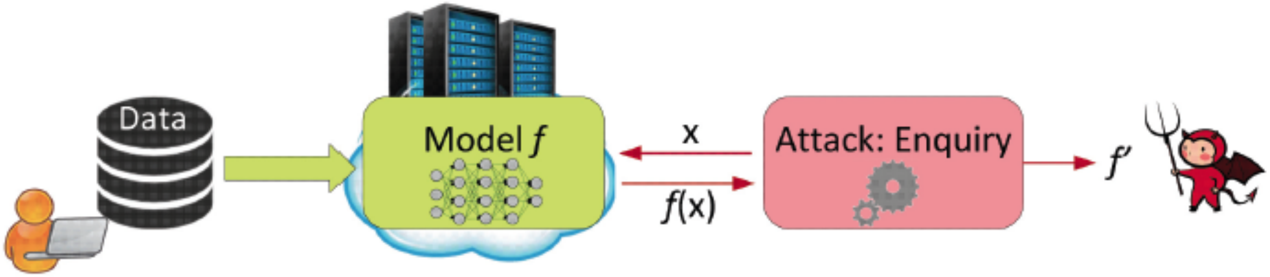
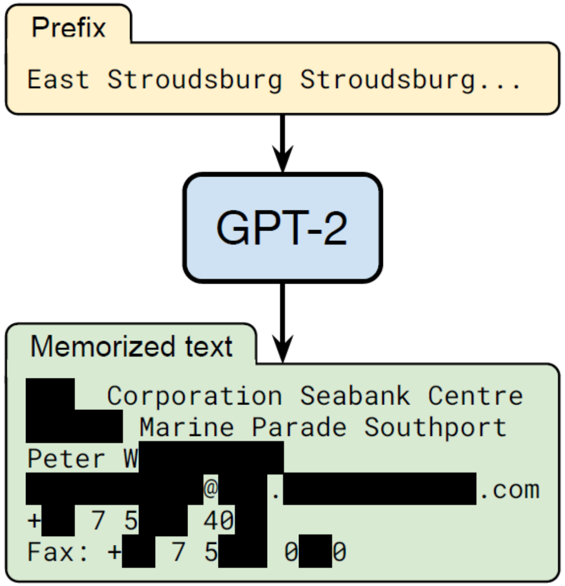


Figure form: Liu et al. (2020) When Machine Learning Meets Privacy: A Survey and Outlook

Training Data Extraction Attack

- Example of training data extraction
  - The authors query GPT-2 by entering the ***prefix***: “East Stroudsburg Stroudsburg…”
  - The model outputted a block of text, which included the full name, phone number, email address, and physical address of the person
  - This information was included in the training data for GPT-2, it was memorized by the model, and extracted by using the training data extraction attack



Prefix

East Stroudsburg Stroudsburg...

GPT-2

Memorized text

[redacted] Corporation Seabank Centre
[redacted] Marine Parade Southport
Peter W[redacted]
[redacted]@[redacted].[redacted].com
+[redacted] 7 5[redacted] 40[redacted]
Fax: +[redacted] 7 5[redacted] 0[redacted]0

Training Data Extraction Attack

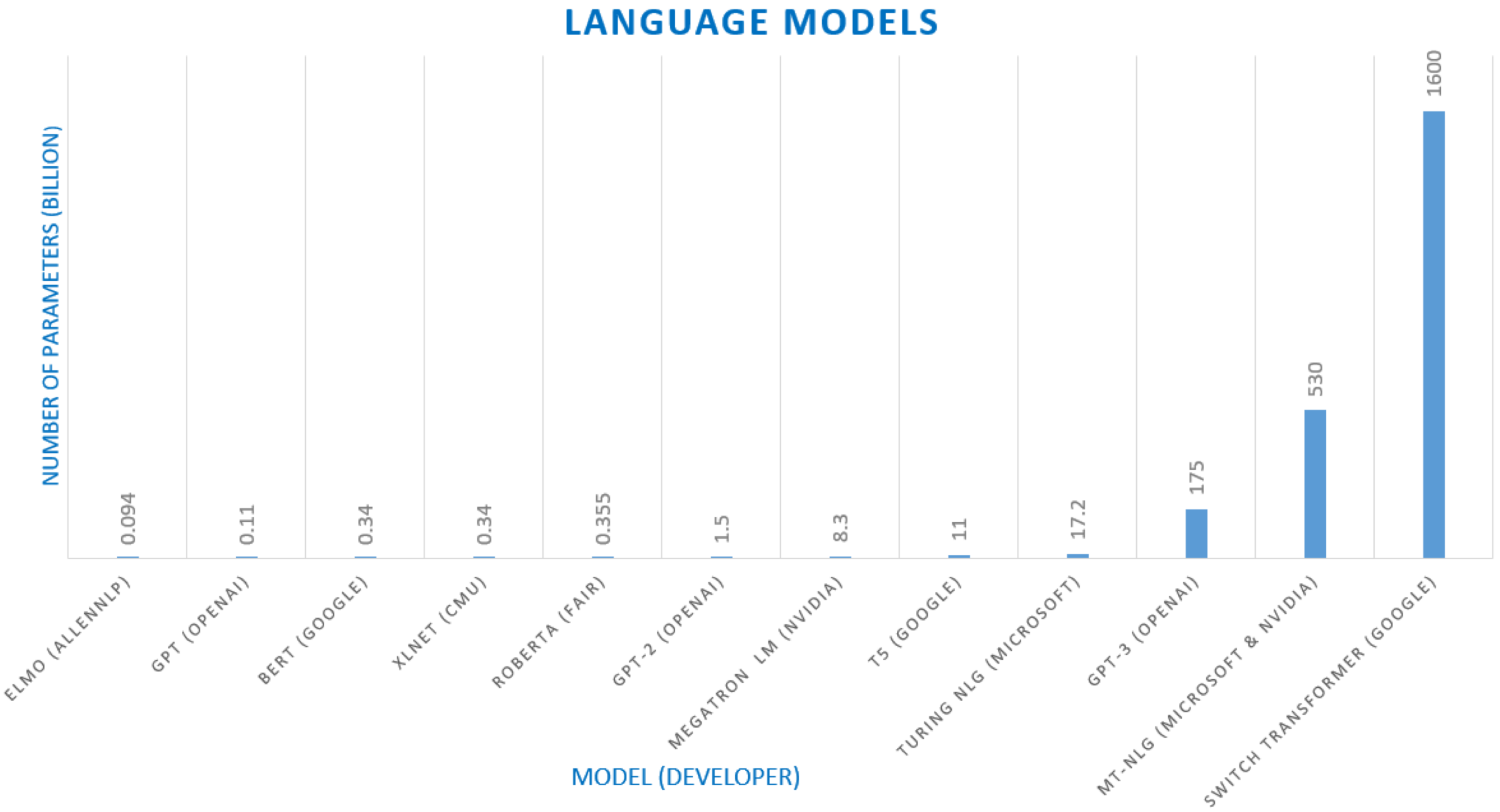
- The main findings of the study are:
  - Most of the memorized samples were found in only 1 document in the dataset
    - However, the samples were repeated multiple times in the document
  - Out of 1,800 candidate sequences that were manually analyzed by the authors, GPT-2 memorized 600 from the public training data
  - Larger language models are more vulnerable to data extraction than smaller models
  - Although LMs are trained on large datasets and therefore they exhibit little overfitting, they can still memorize the training data
- Implications:
  - Training data extraction attacks have previously been limited to small LMs trained on small datasets
  - It was also believed that LMs do not memorize the data, because they exhibit little overfitting
  - Recent LMs are increasingly larger, thus, such vulnerabilities can become more significant

Language Models

- ***Language models (LMs)*** are statistical models that assign a probability to a sequence of words
- Modern language models include:
  - Switch Transformer (Google, 2021): 1.6 trillion parameters
  - MT-NLPG or Megatron Turing NLG (Microsoft & NVIDIA, 2021): 530 billion parameters
  - GPT-3 (OpenAI, 2020): 175 billions parameters
  - Turing NLG, or Natural Language Generation (Microsoft, 2020): 17 billion parameters
  - T5, or Text-to-Text Transfer Transformer (Google, 2019): 11 billion parameters
  - Megatron ML (NVIDIA, 2019): 8.3 billion parameters
  - GPT-2 (OpenAI, 2019): 1.5 billion parameters
  - BERT, or Bidirectional Encoder Representations from Transformers (Google, 2018): 110 million parameters
  - GPT (OpenAI, 2018): 110 million parameters

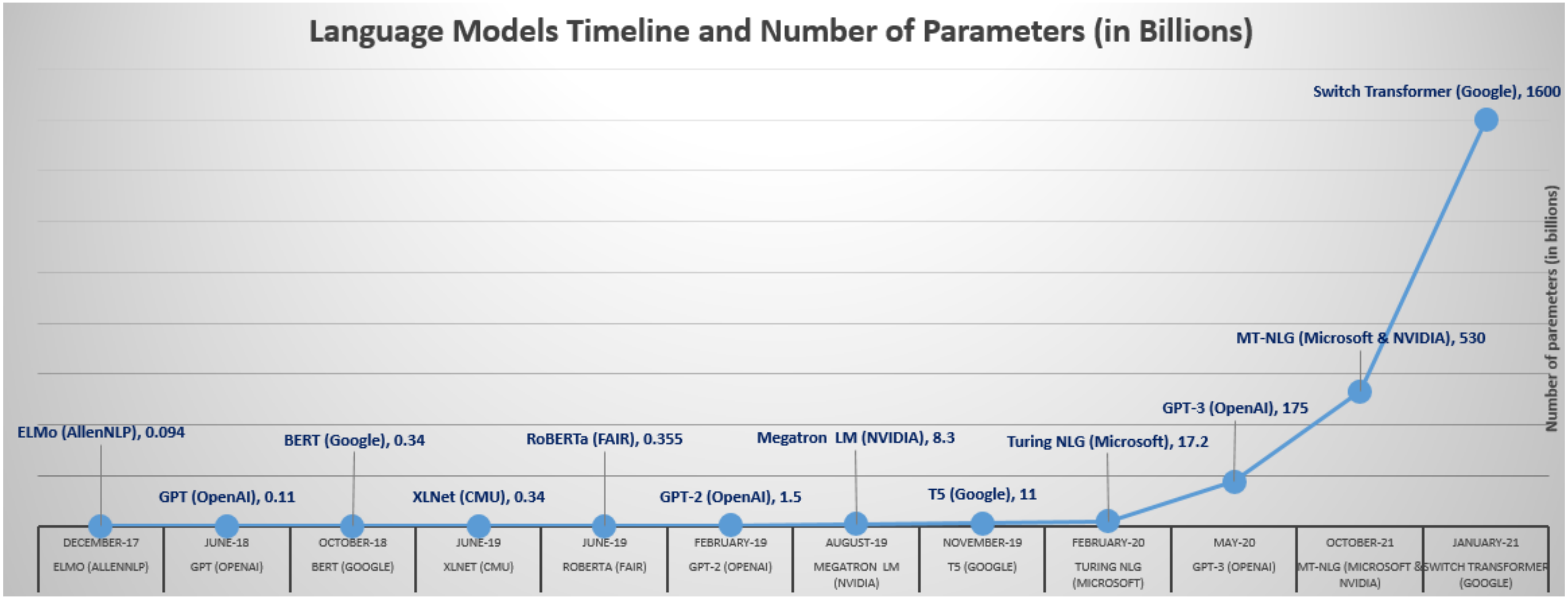
Language Models Graph

- Language models and number of parameters



Language Models Timeline

- Language models timeline



Language Models

- Language models are trained on large training datasets containing raw text scraped from the Internet
  - LM generate text sequences as fluent natural language
    - Therefore, are generative ML models
  - The quality of text generated by LMs is often undistinguishable from human-written text
- A common approach to train LMs is the "next-step prediction" objective
  - I.e., when prompted with a sequence of words, predict the next word in the sequence
  - This is unsupervised learning, since the training set is not labeled
- Given a sequence of tokens (word embeddings) $x_1, x_2, \ldots, x_{i-1}$ from a vocabulary $\mathcal{V}$, the objective is to estimate the probability of the next token x_i in the sequence given the previous tokens, i.e., $\mathcal{P}(x_i | x_1, x_2, \ldots, x_{i-1})$
  - E.g., given the sequence "Marry had a little," the word "lamb" is the most likely next word in the sequence

Language Models

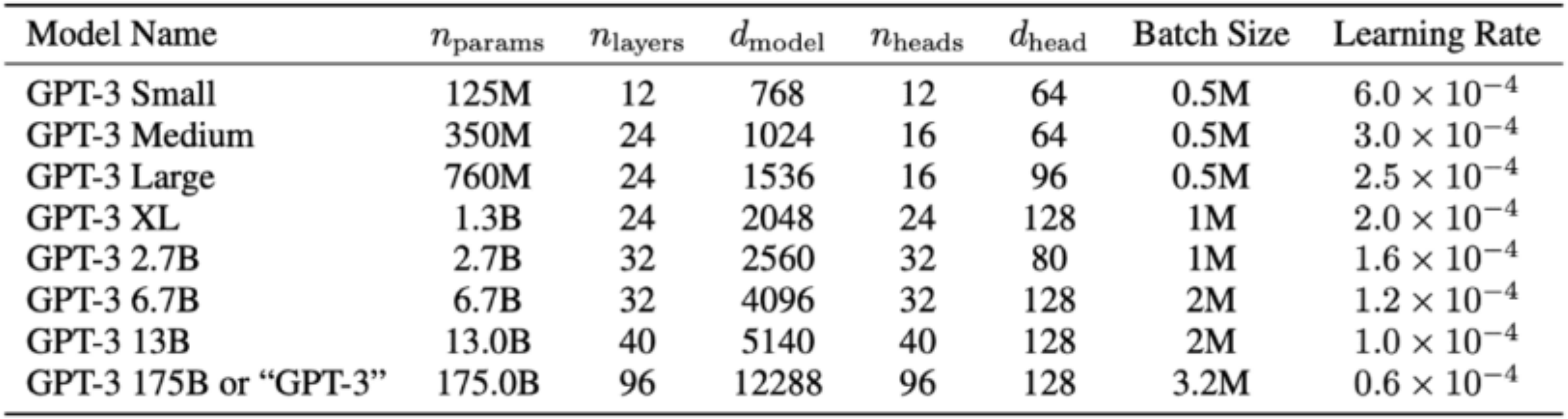
- NNs are currently used for estimating the likelihood of the next token x_i in NLP
  - Earlier works in NLP used architectures with recurrent layers (e.g., LSTM layers)
  - Recent works use attention-based architectures
    - *Transformer* is one such model that has been regularly used in recent LMs, consisting of a sequence of attention layers
- Training examples in LMs are text documents
  - E.g., webpages, new articles from the Internet
- The training loss minimizes the error in predicting the next token
  - For an NN f with parameters θ, the objective is to find network parameters θ that minimize the loss $\mathcal{L}(\theta) = -\log \prod_{i=1}^{n} f_\theta(x_i | x_1, x_2, \dots, x_{i-1})$
  - n is the number of tokens in the vocabulary $\mathcal{V}$

GPT-2

- *GPT-2* (Generative Pre-training Transformer) was released by OpenAI in 2019
- It is a family of several models with varying number of parameters, trained using the same dataset
    - GPT-2 XL: 1.5 billion parameters
    - GPT-2 Medium: 334 million parameters
    - GPT-2 Small: 124 million parameters
- The training dataset consists of 40GB of de-duplicated text data
    - The data is scraped from publicly available sources from the Internet
- For the training data extraction attack the authors used GPT-2 XL
    - It was found that GPT-2 XL memorized 10 times more information than GPT-2 Small

GPT-3

- *GPT-3* was released by OpenAI in 2020
  - Training dataset: 45 TB of text
  - The cost for training GPT-3 175B reportedly is $US 12 million
- Besides sentence continuation based on user's prompt, GPT-3 has also been used for other tasks, such as answering questions, summarizing text, language translation, information retrieval, automated code generation, etc.
  - Check an example of generating JavaScript XML (JSX) code for web and mobile User Interface applications by GPT-3 (link) and OpenAI's Codex model (link)
- T



Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or "GPT-3"	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

Language Models

- LMs in other languages have also been developed recently
  - Wudao (Chinese LM, 1.75 trillion parameters), HyperCLOVA (Korean)
  - The German AI startup Aleph Alpha developed a large European language model, fluent in English, German, French, Spanish, and Italian
- Microsoft recently trained ZeRO-Infinity LM with 32 trillion parameters (it is not known if it outperforms current LMs)
- Several LMs are available through black-box API
  - Users can enter a prompt and obtain next-word predictions
- EleutherAI open-sourced a 6 billion parameter model called GPT-J (demo link)
  - It is available from Hugging Face
  - Hugging Face developed open-source NLP library based on the Transformers architectures
- Controversy about LMs:
  - LMs can generate fake news, that are difficult to distinguish from real news

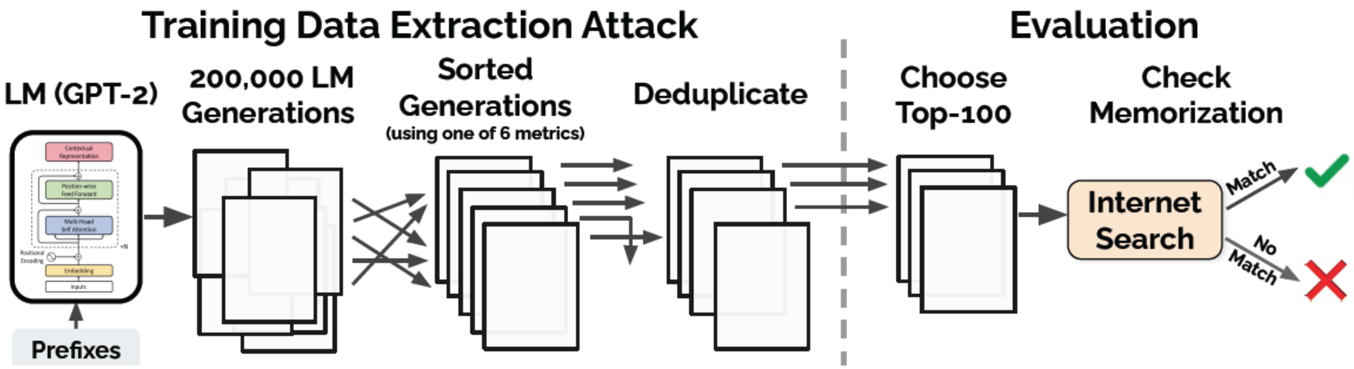
Slide Credit: State of AI Report 2021

Attack Threat Model

- The authors had black-box query access to GPT-2
- Objective: extract memorized training data
  - The strength of the attack is measured based on the number of documents in which the text appeared
    - Memorizing one word that occurred in many training examples (documents) is not severe
  - Stronger attack extract text that occurred in one single document
    - This is referred to as "unintended" memorization
- The training data for GPT-2 was collected by OpenAI from public sources
  - OpenAI didn't release the training dataset
    - But they released a document on the data collection process
  - The authors downloaded the public data by following the documentation
    - They didn't have access to the actual dataset used by OpenAI

Attack Approach

- Attack approach
  - Generate many text samples by using prefix prompts to GPT-2
    - Build 3 datasets of 200,000 generated samples, each sample if 256 tokens long
  - Sort the generated output text using 6 metrics (see next page)
    - LM has high confidence when the text is taken directly from the training data
  - Remove duplicate outputs
  - For the top 100 outputs, perform an Internet search to confirm whether the generated text is an exact match to a web document
  - Check with OpenAI to confirm if the extracted text occurred in their training dataset

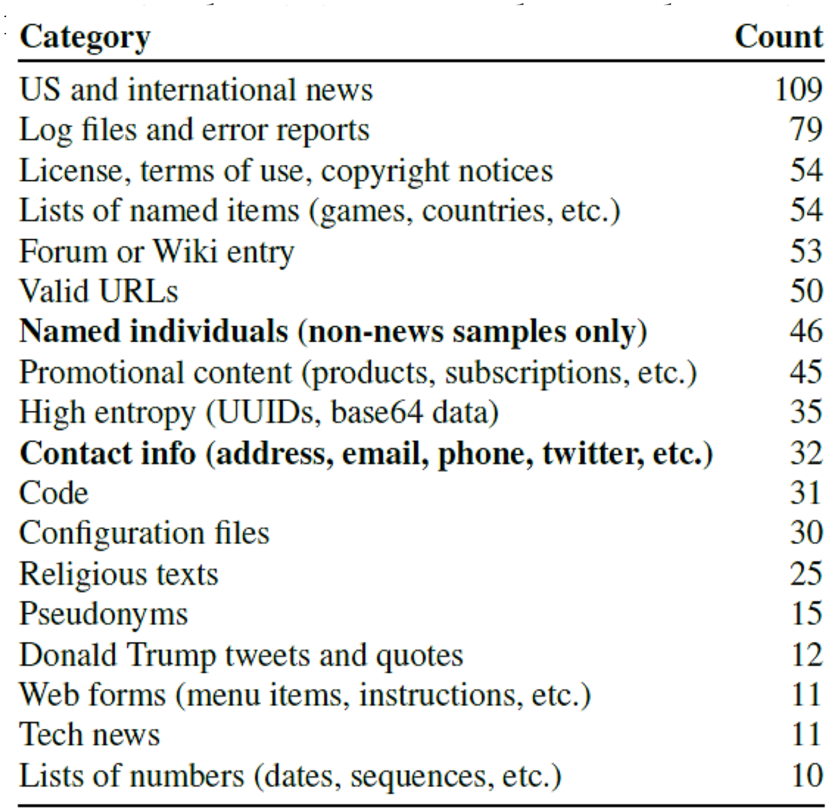


Attack Approach

- *Metrics for sorting* the predicted text
  1. Perplexity, quantifies the level of "surprise" by the GPT-2 model
     - Defined as $P = \exp(-(1/n)\sum_{i=1}^{n} \log f_\theta(x_i | x_1, x_2, \ldots, x_{i-1}))$
  2. Comparison to the predictions by GPT-Small and GPT-Medium models
     - It is less likely that the different models will memorize the same data
  3. Text entropy when the output is compressed using zlib compression
  4. Perplexity when the text is switched from uppercase to lowercase letters
  5. Averaged perplexity using a sliding window of 50 tokens
- Other strategies to improve the attack:
  - Use prompts based on Internet text
- The authors used 3 datasets of 200,000 generated samples
  - For the 6 metrics above, this resulted in 3×6 configurations, or 1,800 top samples

Results

- The authors manually inspected 1,800 generated samples from GPT-2
- They identified 604 memorized training examples (about 33% of the samples)
  - The categories of the table



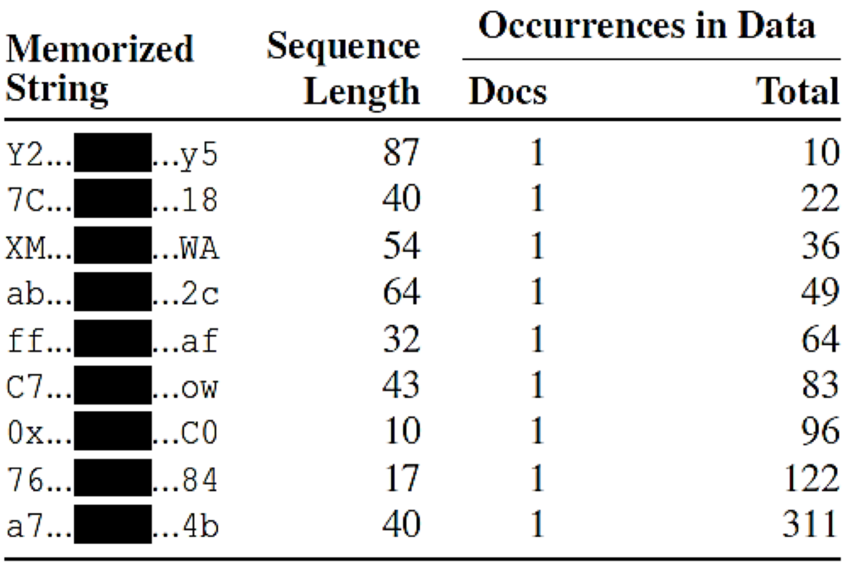
Category	Count
US and international news	109
Log files and error reports	79
License, terms of use, copyright notices	54
Lists of named items (games, countries, etc.)	54
Forum or Wiki entry	53
Valid URLs	50
Named individuals (non-news samples only)	46
Promotional content (products, subscriptions, etc.)	45
High entropy (UUIDs, base64 data)	35
Contact info (address, email, phone, twitter, etc.)	32
Code	31
Configuration files	30
Religious texts	25
Pseudonyms	15
Donald Trump tweets and quotes	12
Web forms (menu items, instructions, etc.)	11
Tech news	11
Lists of numbers (dates, sequences, etc.)	10

Results

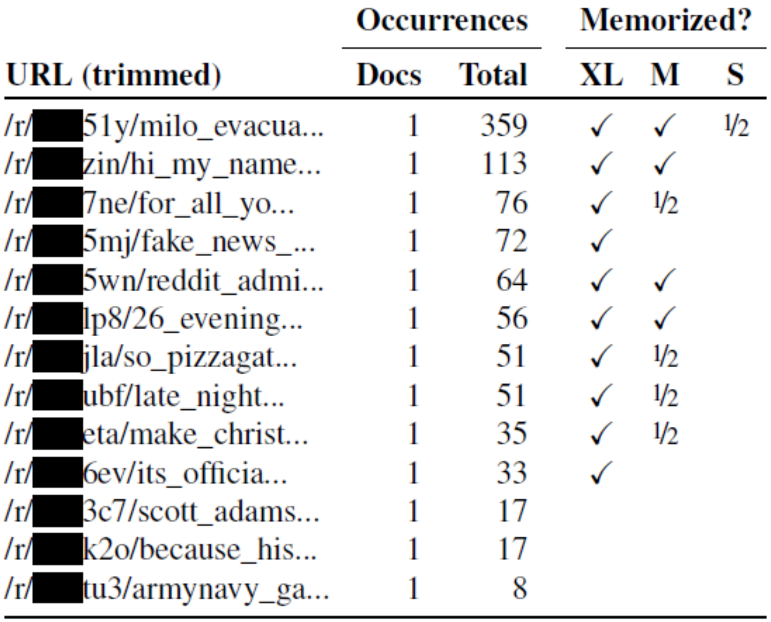
- Examples of memorized content
  - Personally identifiable information (PII)
    - Found 78 examples of peoples' names, phone numbers, addresses, and social media accounts
    - E.g., extracted the usernames of participants in an Internet forum conversation that appeared in one training document
  - URLs
    - 50 examples of memorized URLs
  - Code
    - Identified 31 samples that contain snippets of memorized source code
  - Unnatural text
    - E.g., UUID: 1e4bd2a8-e8c8-4a62-adcd-40a936480059
- GPT-2 memorized removed content
  - The authors extracted content that has been removed from the Internet, but it had been memorized by GPT-2

Results

- Examples of memorized strings of unnatural text and URLs that occurred in only 1 document in the training data
  - E.g., in the left table the first extracted string listed has 87 characters, and occurred 10 times in 1 document
  - In the right table, we can see that GPT-XL model memorized more content than GPT-S and GPT-M models



Memorized String	Sequence Length	Occurrences in Data	
		Docs	Total
Y2...[redacted]...y5	87	1	10
7C...[redacted]...18	40	1	22
XM...[redacted]...WA	54	1	36
ab...[redacted]...2c	64	1	49
ff...[redacted]...af	32	1	64
C7...[redacted]...ow	43	1	83
0x...[redacted]...C0	10	1	96
76...[redacted]...84	17	1	122
a7...[redacted]...4b	40	1	311



	Occurrences		Memorized?		
URL (trimmed)	Docs	Total	XL	M	S
/r/[redacted]51y/milo_evacua...	1	359	✓	✓	½
/r/[redacted]zin/hi_my_name...	1	113	✓	✓	
/r/[redacted]7ne/for_all_yo...	1	76	✓	½	
/r/[redacted]5mj/fake_news_...	1	72	✓		
/r/[redacted]5wn/reddit_admi...	1	64	✓	✓	
/r/[redacted]lp8/26_evening...	1	56	✓	✓	
/r/[redacted]jla/so_pizzagat...	1	51	✓	½	
/r/[redacted]ubf/late_night...	1	51	✓	½	
/r/[redacted]eta/make_christ...	1	35	✓	½	
/r/[redacted]6ev/its_officia...	1	33	✓		
/r/[redacted]3c7/scott_adams...	1	17			
/r/[redacted]k2o/because_his...	1	17			
/r/[redacted]tu3/armynavy_ga...	1	8			

Mitigation Strategies for Data Leakage

- ***Selecting the training data*** for LMs
  - Avoid text from websites that are known to host sensitive data
  - Apply methods that limit the amount of sensitive content
    - E.g., filter personal information
  - De-duplicate content in the training data
- ***Training with differential privacy***
  - DP can reduce, but cannot prevent, memorization of content that occurs often in the dataset
  - Limitation: reduced accuracy, longer training times
- ***Auditing LMs*** for memorization
  - Determine the level of memorization in LMs
  - E.g., the training data extraction attack can be used to evaluate the level of memorization of an LM

Ethical Considerations and Lessons

- The authors contacted the individuals whose PII was extracted and obtained permissions to include it in the paper
  - All PII in the paper is masked with a black rectangle box
- Among the 600,000 generated samples, 604 (or 0.1%) contain memorized text
  - The authors manually inspected only 1,800 samples
- For complete memorization, it was estimated that the content should occur 33 times in one single document
- It is important to further study and understand memorization in LMs, and develop prevention strategies