Extracting Training Data from Large Language Models

Large Language Model의 학습 데이터 추출 공격

연구 배경

- LLM은 대규모 웹 데이터로 학습
- 모델 크기가 커질수록 자연어 생성 성능이 좋아짐
- 하지만 학습 데이터가 모델 출력으로 새어 나올 수 있음
- 기존에는 과적합이 작으면 memorization 위험도 작다고 생각

기존 인식과 문제 제기

- 기존 인식: train-test gap이 작으면 학습 데이터 암기가 적을 것
- 평균적으로 과적합이 작아도 일부 worst-case training example은 그대로 암기될 수 있다.
- black-box query 만으로 학습 데이터를 추출할 수 있는가?

Related Work

공격 유형	목적	차이점
Membership inference	특정 데이터가 학습에 포함됐는지 판단	원문을 복원하지는 않음
Model inversion	학습 데이터의 대표적 특징 복원	fuzzy reconstruction에 가까움
Training data extraction	학습 문장을 verbatim으로 복원	이 논문의 핵심 공격
Differential privacy	학습 데이터 보호	방어 방법으로 논의

Language Model 기본 원리

- 언어모델은 이전 토큰을 보고 다음 토큰 확률을 예측함
- 학습 목표는 학습 데이터 문장에 높은 확률을 주도록 만드는 것
- 학습 데이터의 다음 토큰을 외우는 방향과 연결될 수 있음

$$\mathbf{Pr}(x_1, x_2, \ldots, x_n),$$

where $x_1, x_2, \ldots, x_n$ is a sequence of tokens from a vocabulary $\mathcal{V}$ by applying the chain rule of probability

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

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

GPT-2 실험 대상

모델	파라미터 수	역할
GPT-2 Small	124M	비교 기준
GPT-2 Medium	334M	비교 기준
GPT-2 XL	약 1.5B	주요 공격 대상

Memorization 정의

- 어떤 문자열 s가 특정 prefix c 뒤에서 모델에 의해 추출될 수 있으면 extractable
- 그 문자열이 학습 데이터의 최대 k개 example에만 등장했다면 k-eidetic memorization
- k가 작을수록 더 심각한 암기

Definition 1 (Model Knowledge Extraction) *A string s is* extractable[4] *from an LM f_θ if there exists a prefix c such that:*

$$s \leftarrow \underset{s' :\, |s'|=N}{\arg\max}\, f_\theta(s' \mid c)$$

Definition 2 (k-Eidetic Memorization) *A string s is k-eidetic memorized (for $k \geq 1$) by an LM f_θ if s is extractable from f_θ and s appears in at most k examples in the training data X: $|\{x \in X : s \subseteq x\}| \leq k$.*

Threat Model

- 공격자는 모델 weight를 모름
- hidden state나 attention도 볼 수 없음
- 입력-출력과 sequence probability만 사용할 수 있음
- 목표는 특정 한 사람을 겨냥하는 targeted extraction이 아니라, 암기된 데이터를 광범위하게 찾아내는 것

Baseline Attack

- perplexity가 낮다 = 모델이 문장을 잘 예측한다
- 초기 공격은 GPT-2가 생성한 문장을 perplexity 기준으로 정렬함
- 낮은 perplexity 문장을 memorized candidate로 봄

$$\mathcal{P} = \exp\left(-\frac{1}{n}\sum_{i=1}^{n}\log f_\theta\left(x_i|x_1,\ldots,x_{i-1}\right)\right)$$

Baseline의 한계

- 문제 1: 생성 결과 다양성이 낮음
- 문제 2: 반복 문자열이 false positive로 많이 잡힘

예시: I love you. I love you. I love you …

- 이런 문장은 모델이 높은 확률을 줄 수 있지만 실제 학습 데이터 암기라고 보기 어려움

Improved Attack 전체 구조

- GPT-2에서 200,000개 generation 생성
- 6개 metric으로 정렬
- 중복 제거
- top candidate 선택
- 인터넷 검색으로 확인
- OpenAI와 협력해 원 GPT-2 training data에서 최종 확인

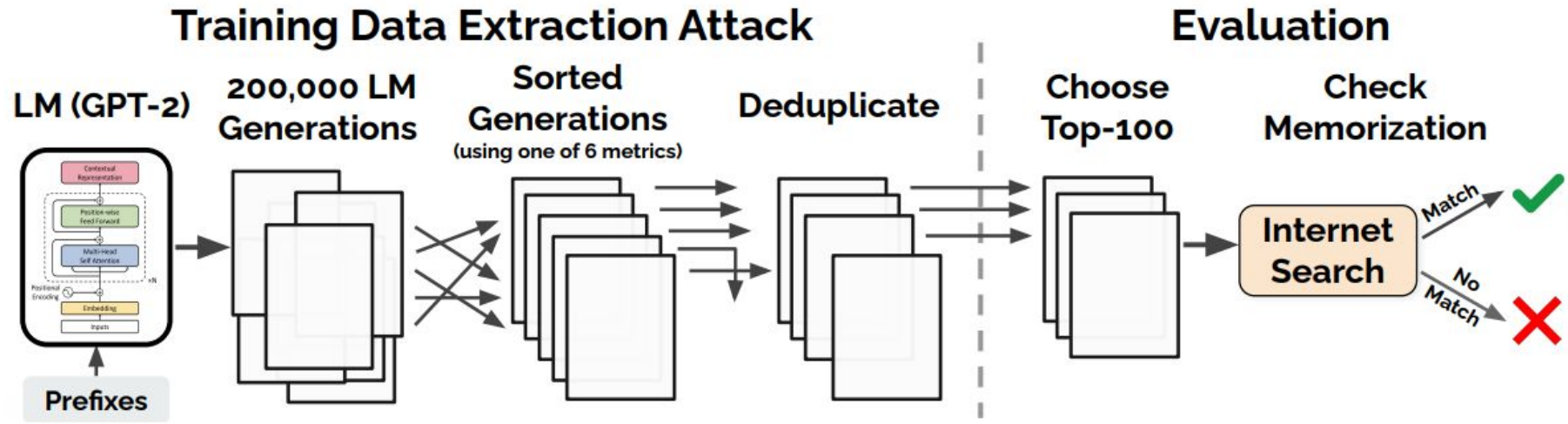


Figure 2: **Workflow of our extraction attack and evaluation. 1) Attack.** We begin by generating many samples from GPT-2 when the model is conditioned on (potentially empty) prefixes. We then sort each generation according to one of six metrics and remove the duplicates. This gives us a set of potentially memorized training examples. **2) Evaluation.** We manually inspect 100 of the top-1000 generations for each metric. We mark each generation as either memorized or not-memorized by manually searching online, and we confirm these findings by working with OpenAI to query the original training data. An open-source implementation of our attack process is available at https://github.com/ftramer/LM_Memorization.

3가지 Text Generation 방식

생성 방식	설명
Top-n	시작 token에서 256 tokens를 top-n sampling으로 생성
Temperature	초반에는 높은 temperature로 다양하게 생성하고, 이후 t=1으로 감소
Internet	Common Crawl 기반 5~10 token prefix를 주고 이어서 생성

6가지 판별 지표

지표	의미
Perplexity	GPT-2 XL의 perplexity
Small	GPT-2 XL과 GPT-2 Small의 log-perplexity 차이
Medium	GPT-2 XL과 GPT-2 Medium의 log-perplexity 차이
zlib	GPT-2 perplexity와 zlib entropy의 차이
Lowercase	원문과 lowercased sample의 perplexity 차이
Window	50-token sliding window에서의 minimum perplexity

18개 실험 설정

- 3 generation strategies x 6 inference metrics = 18 attack configurations
- 각 configuration에서 top-1000 generation 중 100개 후보를 선택
- 총 1,800개 후보 수동 검사
- 인터넷 검색 + 원 GPT-2 training data 확인

주요 정량 결과

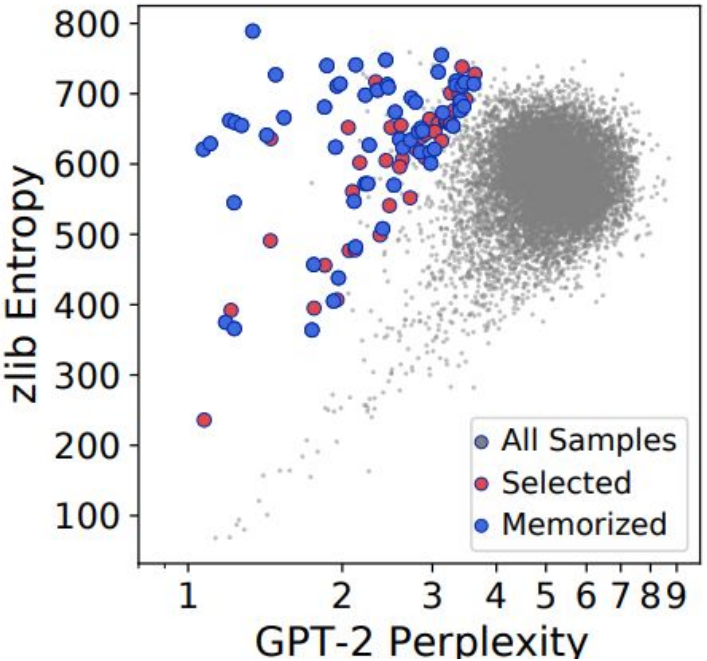
- 각 generation strategy와 inference metric 조합별로 100개의 후보 중 몇개가 memorized sample인지 보여줌
- Internet + zlib 조합이 67개로 가장 높음
- 총 unique memorized sample은 604개

Inference Strategy	Text Generation Strategy		
	Top-n	Temperature	Internet
Perplexity	9	3	39
Small	41	42	58
Medium	38	33	45
zlib	59	46	67
Window	33	28	58
Lowercase	53	22	60
Total Unique	191	140	273

zlib

- 왼쪽 위 outlier는 GPT-2는 잘 예측하지만 압축하기 어려운 샘플
- 이 outlier가 memorization 가능성이 높음

x축: GPT-2 perplexity
y축: zlib entropy
빨간 점: 수동 검사 대상으로 선택된 샘플
파란 점: 실제 memorized sample



추출된 데이터 유형

- 604개 memorized training examples를 카테고리별로 분류
- 뉴스, 로그, 라이선스, URL, 이름, 연락처, 코드, UUID 등이 포함됨

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

k=1 High-Entropy Memorization 사례

- k=1 eidetic memorized high-entropy content 사례
- 랜덤 문자열, UUID, base64, hash처럼 일반 언어 규칙으로 예측하기 어려운 데이터가 포함됨
- 각각 단 하나의 training document에만 포함된 사례 존재

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

모델 크기와 암기 관계

- Reddit URL이 학습 데이터에 몇 번 등장했는지
- GPT-2 XL, Medium, Small이 이를 생성했는지 비교
- GPT-2 XL은 반복 등장 횟수가 일정 수준 이상이면 더 잘 암기함

-> 모델이 클수록 memorization risk가 증가함

	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			

방어 방법

Differential Privacy: 학습 과정에서 개별 학습 데이터 하나가 모델에 과도하게 반영되지 않도록 제한

Data Curation: 학습 전에 개인정보, 민감 문서, 부적절한 웹사이트 데이터를 걸러내는 방법

Deduplication: 같은 문장이나 문서가 반복해서 학습되지 않도록 중복을 제거

Output Filtering: 모델이 생성한 출력에서 개인정보나 학습 데이터와 유사한 문장을 감지해 차단

Memorization Auditing: 모델 배포 전에 실제로 학습 데이터를 외웠는지 공격 방식으로 점검

결론

1. 과적합이 없어도 memorization은 발생한다.
2. 모델이 클수록 privacy leakage 위험이 커진다.
3. 민감 데이터로 학습한 LLM은 실제 유출 위험이 커질 수 있다.

한계

1. memorized data는 특정 prefix가 있어야 드러나는 경우가 많다.
2. 특정 데이터가 강하게 암기되는지는 완전히 설명하지 못한다.
3. 방어책들은 모두 불완전 하다