"next-step prediction" objective

generative model: $Pr(x_1, x_2, x_3 \cdots x_n) = Pr(x_1) \times Pr(x_2|x_1) \times Pr(x_3|x_1, x_2) \cdots$

$= \prod_{i=1}^{n} Pr(x_i|x_1, \cdots x_{i-1})$

$f_\theta(x_i|x_1, x_2 \cdots x_{i-1})$: the likelihood of token x_i based on
- the neural networks f
- the parameters θ

for state of art. LMs: the training loss and the test loss is nearly identical
GPT-2 in this paper.

No significant overfitting.

Defining Language Model Memorization

Eidetic Memory : model memorized this part of data though only appearing in a small set of training instances.

Have knowledge of sting s: if s can be extracted by interacting with the model:

model f_θ knows the string s.

According to the generation of string s with model f_θ:

$f_\theta(x_i|x_1, x_2 \cdots x_{i-1})$.

"knows string s" : $s \leftarrow \underset{s':|s'|=N}{\arg\max} f_\theta(s'|c)$ (prefix)

k-Eidetic Memorization : s is extractable from f_θ and s appears in at most k training examples.

$|\{x \in X : s \sqsubseteq x\}| \leq k$

Strength of attack can be measured: strength ↑ => the number and length of sequence ↑

k ↓

extract more data with less repeat

Training Data Extraction Attack

1. generate text the start token: $ GPT-2: $f_\theta(x_i|x_1, x_2 \cdots x_{i-1})$.

argmax: top-n samples n=40

Iteration 1: $s_1 \leftarrow \arg\max_{s':|s'|=N} f_\theta(s'|\underbrace{\$}_{\text{prefix}})$

Iteration 2: $s_2 \leftarrow \arg\max_{s':|s'|=N} f_\theta(s'|\underbrace{\$s_1}_{\text{prefix}})$

⋮

Iteration n: $s_n \leftarrow \arg\max_{s':|s'|=N} f_\theta(s'|\underbrace{\$s_1 s_2 \cdots s_{n-1}}_{\text{prefix}})$

"aimlessly" generate text based on the model's memory

2. predit which text contains memorized text Membership Inference.

How to figure out memorized text?

Figure out whether sample is in training data.

How to know whether sample is in training data.?

model tend to assign higher confidence to examples in training data.

Text with highest confidence is likely to be in the training data.

How to measure "confidence"?

High confidence => the sequence is "good"

Measure "good": PPL

According to the principle of probabilistic language model.

Low PPL => High probablity of generating the sequence => Model have seen it once.

Two Weaknesses.

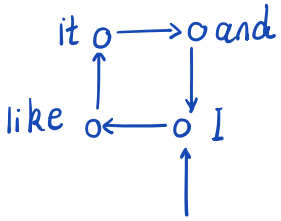
low diversity of output (generate text) sample duplicate

Large number of false positives.(predit which text contains memorized text)

low PPL. but no actuall meaning. repeated strings

repetitive utterance generally get low perplexity. the looping n-gram

https://web.stanford.edu/class/cs224n/slides/cs224n-2023-lecture10-nlg.pdf



arxiv : 2012.14660.pdf.

Improved Training Data Extraction Attack

low diversity of output:

Cause: input always start with $ => duplicate sample.

Resolve: 1. Sampling with a decaying temperature

The output of $Pr(x_i | x_1, x_2 \cdots x_{i-1})$:

$z = f_\theta(x_1, \cdots x_{i-1})$

$Pr = y = softmax(z)$ $softmax(z)_i = \frac{\exp(z_i)}{\sum_{j=1}^{n} \exp(z_j)}$ calculating the weight of each z_i

replace the $softmax(z)$ with $softmax(\frac{z}{t})$ for $t > 1$ t for "temperature"

"flatten" the distribution.

high temperature causes the model to be less confident and more diverse on output.

A decaying temperature: from t=10 at the begining to t=1 at the end:
more diverse at the begining and more confident with the generating
explore with diversity and follow a confident path.

2. generate with a seed prefix choosen from Internet.
choose datasets → generate context seeds → generate text.

Improved Membership Inference.

Large number of false positives.

false positive
- trival memorization numbers from 1~100
- repeat.

high likelihood on all models.

Filtering method: use another LM. smaller GPT-2
filter the sample with unexpected high likelihood on the first model.
examples generated by memory

use zlib compression
the entropy of the text (trival memorization and repeat. are.
with low entropy)

lowercased Text
the PPL of lowercased text is different from the original one.
model can't understand the meaning. it just remember the format.
useful for memory text that are case-sensitive.

PPL on slide window.
useful for high PPL — low PPL - high PPL

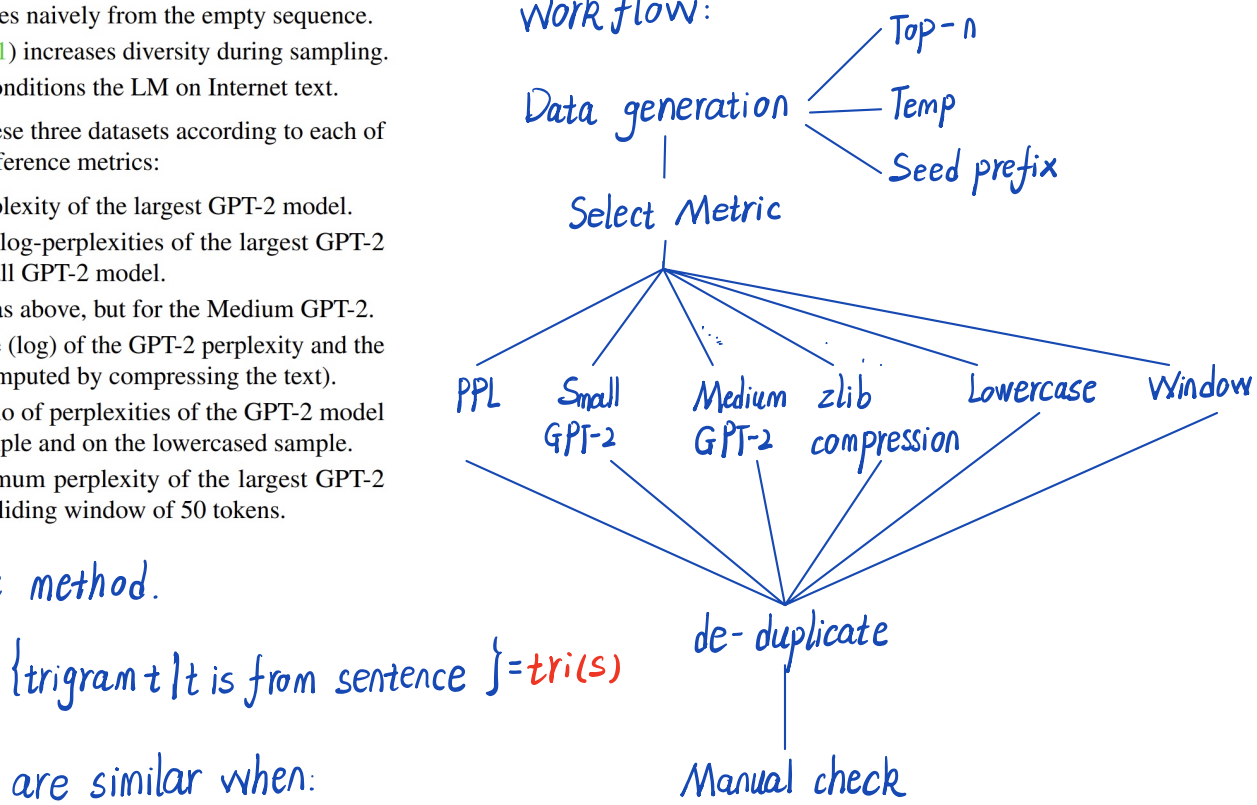
An overview of our experimental setup is shown in Figure 2. We first build three datasets of 200,000 generated samples (each of which is 256 tokens long) using one of our strategies:

- *Top-n* (§4.1) samples naively from the empty sequence.
- *Temperature* (§5.1.1) increases diversity during sampling.
- *Internet* (§5.1.2) conditions the LM on Internet text.

We order each of these three datasets according to each of our six membership inference metrics:

- *Perplexity*: the perplexity of the largest GPT-2 model.
- *Small*: the ratio of log-perplexities of the largest GPT-2 model and the Small GPT-2 model.
- *Medium*: the ratio as above, but for the Medium GPT-2.
- *zlib*: the ratio of the (log) of the GPT-2 perplexity and the zlib entropy (as computed by compressing the text).
- *Lowercase*: the ratio of perplexities of the GPT-2 model on the original sample and on the lowercased sample.
- *Window*: the minimum perplexity of the largest GPT-2 model across any sliding window of 50 tokens.

3x6 =18 configuration



de-duplicate method.

sentense $\longrightarrow$ {trigram t | t is from sentence } = tri(s)

two sentense are similar when:

$$|tri(s_1) \cap tri(s_2| \geq \frac{|tri(s_1)|}{2}$$

How many times must an example appear

Before the model can remember it

Using natural canary in data

prefix -> text generation -> find and count.

results: Larger models remember more

complete memorization is easy for large LM (33times)

How to Mitigate?

Differential Privacy user level differential privacy?

Check the training data.
- a first line of defense
- can't prevent

- remove sensitive content
- de-duplicate: reduce the repeating times.
- carefully source the data

Limiting impact of downstream applications. fine-tuning on task datasets.
How memory is inherited by fine-tuned models.

Auditing Model for Memorization.

Lessons and future work

better prefix can elicit more memory

how to adopt the mitigate method