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

Radek Bartyzal

GLAMI AI

24. 11. 2020

Motivation

Paper is by Google Brain, BAIR.

Current state:

- we are training large NLP models
- scraping a lot of data
- possibly confidential user data

Questions:

- can we extract e.g. card numbers (yes)
- is it due to overfitting? (no)
- how to quantify it? (exposure metric)
- Is my model likely to memorize and potentially expose rarely-occurring, sensitive sequences in training data?

Overview

Figure: There is an XKCD for everything [2].

Threat model

Threat model:

- black box attack
- 1000s of queries
- sees logits / probabilities of the model outputs = it's harder without this

No Transformers?

They only test LSTMs and qRNNS not Transformers!

Methodics

Is my model likely to memorize and potentially expose rarely- occurring, sensitive sequences in training data?

Answer:

- insert randomly-chosen **canary** sequence into training data varying number of times
- how much models memorize = our **exposure metric**
- **exposure**: relative difference in perplexity between canaries and equivalent, non-inserted random sequences
- perplexity $= 2^{H(sequence)}$

Overview

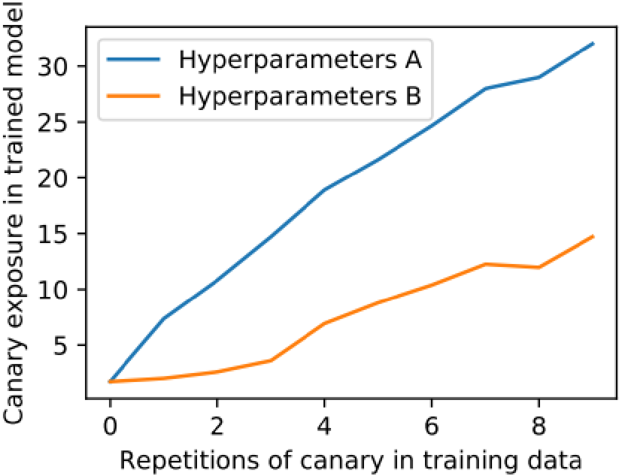

Figure: SOTA word-level language model trained to same accuracy with different hyperparams has very different exposure. If the canary occurs 9 times, it can be extracted from model A.

What are secrets?

- NNs memorize some training data, thats ok if it helps to generalize
- Unintended Memorization = memorize useless data = secrets
- secret = represented by canary sequence
- canary = independent, random sequences from the training data
- $\implies$ canaries are useless for generalization
- $\implies$ insert canaries into training data
- $\implies$ evaluate their exposure in the trained model

Unintended Memorization

When trained neural networks may reveal the presence of out-of-distribution training data.

Perplexity of a sequence

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

$$\begin{aligned}
\mathrm{Px}_\theta(x_1 \ldots x_n) &= -\log_2 \mathbf{Pr}(x_1 \ldots x_n | f_\theta) \\
&= \sum_{i=1}^{n} \left(-\log_2 \mathbf{Pr}(x_i | f_\theta(x_1 \ldots x_{i-1})) \right)
\end{aligned}$$

Exposure metric

- canary = sequence of 9 numbers not in training data
- candidates = other random sequences equal to canary = other 9 numbers that are not in training data
- exposure = $log(rank(canary))$
- $rank(canary)$ = position among candidates ranked by perplexity

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

Estimating exposure = rank of canary

How to est. without calculating perplexity of all (10^9) candidates?

- sample some candidates
- fit skewed normal D over them
- calc. prob. of candidate perplexity $\leq$ canary perplexity

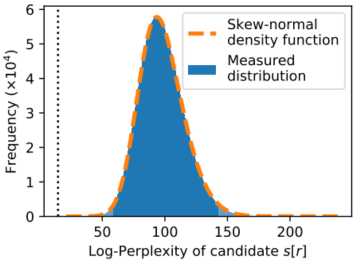

Figure: Skew normal fit to the measured perplexity distribution. The dotted line indicates the log-perplexity of the inserted canary, which is more likely (i.e., has lower perplexity) than any other candidate canary.

Sources

1. Carlini, Nicholas, et al. "The secret sharer: Evaluating and testing unintended memorization in neural networks." 28th USENIX Security Symposium (USENIX Security 19). 2019.
https://arxiv.org/abs/1802.08232

2. XKCD https://xkcd.com/2169/

3. BAIR Blog post.
https://bair.berkeley.edu/blog/2019/08/13/memorization/