Stealing Part of a Production Language Model

Nicholas Carlini [1] Daniel Paleka [2] Krishnamurthy (Dj) Dvijotham [1] Thomas Steinke [1] Jonathan Hayase [3]
A. Feder Cooper [1] Katherine Lee [1] Matthew Jagielski [1] Milad Nasr [1] Arthur Conmy [1] Eric Wallace [4]
David Rolnick [5] Florian Tramèr [2]

It introduces the first model-stealing attack against black-box LLM.

It recovers the exact hidden dimension size of the gpt-3.5-turbo model, and estimate it would cost under $2,000 in queries to recover the entire projection matrix.

Table 1. Summary of APIs

API	Motivation
All Logits §4	Pedagogy & basis for next attacks
Top Logprobs, Logit-bias §5	Current LLM APIs (e.g., OpenAI)
No logprobs, Logit-bias §6	Potential future constrained APIs

Extraction Attack for Logit-Vector APIs

In this section, we assume the adversary can directly view the logits that feed into the softmax function for every token in the vocabulary (we will later relax this assumption), i.e.,

$$\mathcal{O}(p) \leftarrow \mathbf{W} \cdot g_\theta(p) \ .$$

Lemma 4.1. *Let* $\mathbf{Q}(p_1, \dots p_n) \in \mathbb{R}^{l \times n}$ *denote the matrix with columns* $\mathcal{O}(p_1), \dots, \mathcal{O}(p_n)$ *of query responses from the logit-vector API. Then*

$$h \geq \mathit{rank}(\mathbf{Q}(p_1, \dots p_n)) .$$

Proof. We have $\mathbf{Q} = \mathbf{W} \cdot \mathbf{H}$, where $\mathbf{H}$ is a $h \times n$ matrix whose columns are $g_\theta(p_i)$ $(i = 1, \dots, n)$. Thus, $h \geq \text{rank}(\mathbf{Q})$. Further, if $\mathbf{H}$ has rank h (with the second assumption), then $h = \text{rank}(\mathbf{Q})$. □

Algorithm 1 Hidden-Dimension Extraction Attack

Require: Oracle LLM $\mathcal{O}$

Initialize n to an appropriate value greater than h
Initialize an empty matrix $\mathbf{Q} = \mathbf{0}^{n \times l}$
for $i = 1$ to n **do**
    $p_i \leftarrow \texttt{RandPrefix}()$ ▷ Choose a random prompt
    $\mathbf{Q}_i \leftarrow \mathcal{O}(p_i)$
end for
$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \leftarrow \text{SingularValues}(\mathbf{Q})$
count $\leftarrow \arg\max_i \log\|\lambda_i\| - \log\|\lambda_{i+1}\|$
return count

Extraction Attack for Logit-Vector APIs

上海科技大学
ShanghaiTech University

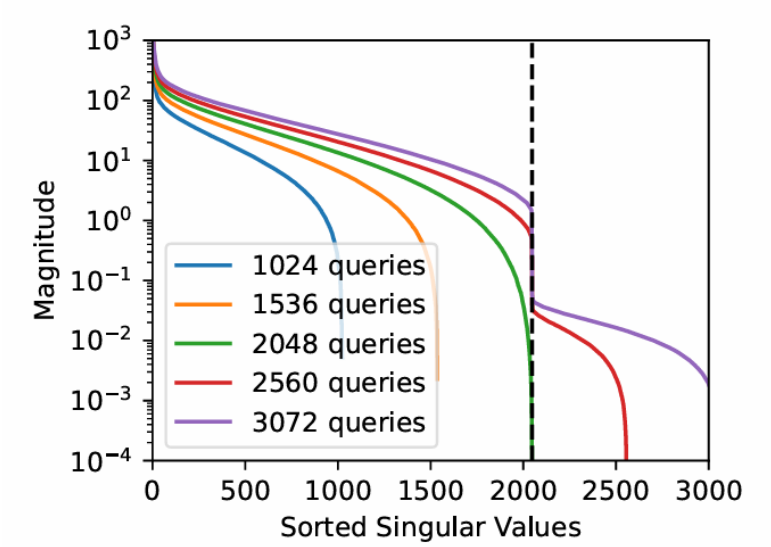


Figure 1. SVD can recover the hidden dimensionality of a model when the final output layer dimension is greater than the hidden dimension. Here we extract the hidden dimension (2048) of the Pythia 1.4B model. We can precisely identify the size by obtaining slightly over 2048 full logit vectors.

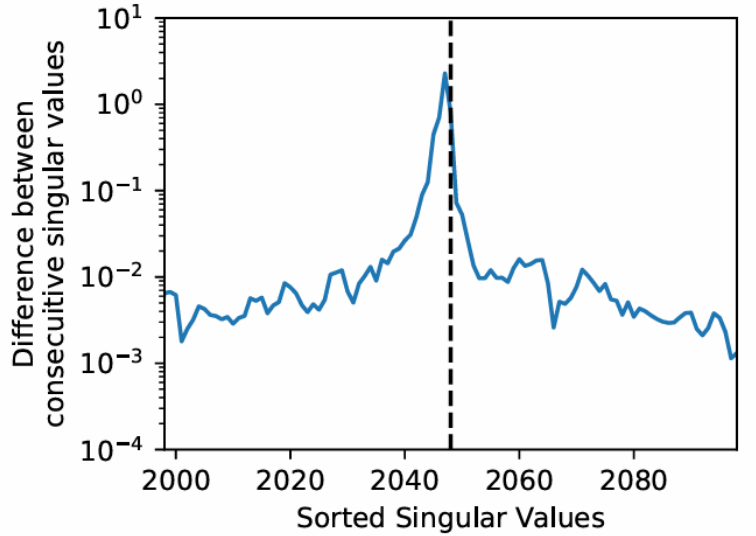


Figure 2. Our extraction attack recovers the hidden dimension size by looking for a sharp drop in singular values, visualized as a spike in the difference between consecutive singular values. On Pythia-1.4B, a 2048 dimensional model, the spike occurs at 2047 singular values.

Extraction Attack for Logit-Vector APIs

上海科技大学
ShanghaiTech University

Lemma 4.2 *In the logit-API threat model, under the assumptions of Lemma 4.1:* (i) *The method from Section 4.2 recovers* $\tilde{\mathbf{W}} = \mathbf{W} \cdot \mathbf{G}$ *for some* $\mathbf{G} \in \mathbb{R}^{h \times h}$*;* (ii) *With the additional assumption that* $g_\theta(p)$ *is a transformer with residual connections, it is impossible to extract* $\mathbf{W}$ *exactly.*

We first give a short proof of (i):

$$\mathbf{Q} = \mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{V}^\top$$

Proof. (i) To show we can recover $\tilde{\mathbf{W}} = \mathbf{W} \cdot \mathbf{G}$, recall Lemma 4.1: we have access to $\mathbf{Q}^\top = \mathbf{W} \cdot \mathbf{H}$ for some $\mathbf{H} \in \mathbb{R}^{h \times n}$. Using the compact SVD of $\mathbf{Q}$ from the method in Section 4.2, $\mathbf{W} \cdot \mathbf{H} \cdot \mathbf{V} = \mathbf{U} \cdot \mathbf{\Sigma}$. We know $\mathbf{G} := \mathbf{H} \cdot \mathbf{V} \in \mathbb{R}^{h \times h}$, hence if we take $\tilde{\mathbf{W}} = \mathbf{U} \cdot \mathbf{\Sigma}$, we have $\tilde{\mathbf{W}} = \mathbf{W} \cdot \mathbf{G}$. □

Experiments. For the six models considered previously, we evaluate the attack success rate by comparing the root mean square (RMS) between our extracted matrix $\tilde{\mathbf{W}} = \mathbf{U} \cdot \mathbf{\Sigma}$ and the actual weight matrix, after allowing for a $h \times h$ affine transformation. Concretely, we solve the least squares system $\tilde{\mathbf{W}} \cdot \mathbf{G} \approx \mathbf{W}$ for $\mathbf{G}$, which reduces to h linear least squares problems, each with l equations and h unknowns. Then, we report the RMS of $\mathbf{W}$ and $\tilde{\mathbf{W}} \cdot \mathbf{G}$.

Table 2. Our attack succeeds across a range of open-source models, at both stealing the model size, and also at reconstructing the output projection matrix (up to invariances; we show the root MSE).

Model	Hidden Dim	Stolen Size	$\mathbf{W}$ RMS
GPT-2 Small (fp32)	768	757 ± 1	$4 \cdot 10^{-4}$
GPT-2 XL (fp32)	1600	1599 ± 1	$6 \cdot 10^{-4}$
Pythia-1.4 (fp16)	2048	2047 ± 1	$3 \cdot 10^{-5}$
Pythia-6.9 (fp16)	4096	4096 ± 1	$4 \cdot 10^{-5}$
LLaMA 7B (fp16)	4096	4096 ± 2	$8 \cdot 10^{-5}$
LLaMA 65B (fp16)	8192	8192 ± 2	$5 \cdot 10^{-5}$

Extraction Attack for Logit-Bias APIs

In this section we develop attacks for APIs that return log probabilities for the top K tokens (sorted by logits), and where the user can specify a real-valued bias $b \in \mathbb{R}^{|\mathcal{X}|}$ (the "logit bias") to be added to the logits for specified tokens before the softmax, i.e.,

$$\mathcal{O}(p, b) \leftarrow \mathsf{TopK}\left(\mathsf{logsoftmax}\left(\mathbf{W}g_\theta(p) + b\right)\right)$$

$$= \mathsf{TopK}\left(\mathbf{W}g_\theta(p) + b - \log\left(\sum_i \exp(\mathbf{W}g_\theta(p) + b)_i\right) \cdot \mathbf{1}\right).$$

$$y_i^B = z_i + B - \log\left(\sum_{j \neq i} \exp(z_j) + \exp(z_i + B)\right)$$

```
completion = client.chat.completions.create(
  model="gpt-3.5-turbo",
  messages=[{"role": "system", "content": "You finish user's sentences."},
            "role": "user", "content": "Once upon a"} ]
  logit_bias={2435:-100, 640:-100}
)
```

We can see log-probs of all tokens after adding bias

Suppose the top token for a prompt is R

$$y_R^B = z_R - \log(\ldots)$$

$$y_R^B - y_i^B - B = z_R + z_i\,.$$

Extraction Attack for Logit-Bias APIs

Token cost: the number of tokens the adversary sends to (or receives from) the model during the attack. Most APIs charge users per-token, so this metric represents the monetary cost of an attack (after scaling by the token cost).

→ $Per\ logit: \frac{1}{2}\ or\ \frac{2+\Delta}{4}(if\ having\ overhead\ tokens)$

Query cost: the total duration of the attack. Most APIs place a limit on the number of queries an adversary can make in any given interval, and so some attacks may be faster but cost more (by sending more tokens per query).

→ $Per\ logit: \frac{1}{4}$

Extraction From Logprob-free APIs

API: Some APIs provide access to a logit bias term, but do not provide any information about the logprobs. Thus, we have,

$$\mathcal{O}(p, b) = \mathsf{ArgMax}\left(\mathsf{logsoftmax}\left(\mathbf{W} \cdot g_\theta(p) + b\right)\right).$$

Given a prompt, let the most likely token be 0.

Algorithm 2 Learning logit differences

$\alpha_i \leftarrow -B, \beta_i \leftarrow 0$
while $\beta_i - \alpha_i > \varepsilon$ **do**
  if $\mathcal{O}\left(p, b = \{i : -\frac{\alpha_i+\beta_i}{2}\}\right) = 0$ **then**
    $\beta_i \leftarrow \frac{\alpha_i+\beta_i}{2}$
  else
    $\alpha_i \leftarrow \frac{\alpha_i+\beta_i}{2}$
  end if
  Return $\frac{\alpha_i+\beta_i}{2}$
end while

Lemma 6.1. *For every token i such that $\mathsf{logit}_i - \mathsf{logit}_0 \geq -B$, Algorithm 2 outputs a value that is at most ε away from the $\mathsf{logit}_i - \mathsf{logit}_0$ in at most $\log\left(\frac{B}{\varepsilon}\right)$ API queries.*

Proof. The API returns the (re-ordered) token 0 as long as the logit bias added is smaller than $\mathsf{logit}_i - \mathsf{logit}_0$. By the assumption, we know that $\mathsf{logit}_i - \mathsf{logit}_0 \in [-B, 0]$. The algorithm ensures that $\beta_i \geq \mathsf{logit}_i - \mathsf{logit}_0 \geq \alpha_i$ at each iteration, as can be seen easily by an inductive argument. Further, $\beta_i - \alpha_i$ decreases by a factor of 2 in each iteration, and hence at termination, we can see that the true value of $\mathsf{logit}_i - \mathsf{logit}_0$ is sandwiched in an interval of length ε. Furthermore, it is clear that the number of iterations is at most $\log_2\left(\frac{B}{\varepsilon}\right)$ and hence so is the query cost of this algorithm. □

Extraction From Logprob-free APIs

- **Improved Logprob-free Attack: Hyperrectangle Relaxation Center**

Algorithm 3 Learning logit differences with multi-token calls

$\alpha_i \leftarrow -B, \beta_i \leftarrow 0 \quad \forall i = 1, \ldots, N$
$\mathcal{C} = \{\text{logit} : \text{logit}_i - \text{logit}_0 \leq B \quad \forall i = 1, \ldots, N\}$
for T rounds **do**
  $b_i \leftarrow -\frac{\alpha_i + \beta_i}{2}$ for $i = 0, \ldots, N$
  $k \leftarrow \mathcal{O}(p, b = \{0 : b_0, 1 : b_1, \ldots, N : b_N\})$
  for $j \neq k$ **do**
    $\mathcal{C} \leftarrow \mathcal{C} \cap \{\text{logit} : \text{logit}_k + b_k \geq \text{logit}_j + b_j\}$
  end for
  for $i = 0, \ldots, N$ **do**
    $\alpha_i \leftarrow \min_{\text{logit} \in \mathcal{C}} \text{logit}_i - \text{logit}_0$
    $\beta_i \leftarrow \min_{\text{logit} \in \mathcal{C}} \text{logit}_i - \text{logit}_0$
  end for
end for
Return $[\alpha_i, \beta_i] \quad \forall i \in \{0, \ldots, N\}$

Lemma 6.2. *Suppose that* $\text{logit}_i - \text{logit}_0 \in [-B, 0]$ *for all* $i = 1, \ldots, l$. *Then, Algorithm 3 returns an interval* $[\alpha_i, \beta_i]$ *such that* $\text{logit}_i - \text{logit}_0 \in [\alpha_i, \beta_i]$ *for each* i *such that* $\text{logit}_i - \text{logit}_0 \in [-B, 0]$. *Furthermore, each round*

$$logit_i - logit_0 \leq B$$

If model still outputs 0 at first rounds:

$$logit_0 + 0 \geq logit_i + B/2$$

$$logit_i - logit_0 \leq -B/2$$

Evaluation

Table 3. Average error at recovering the logit vector for each of the logit-estimation attacks we develop. Our highest precision, and most efficient attack, recovers logits nearly perfectly; other attacks approach this level of precision but at a higher query cost.

Attack	Logprobs	Bits of precision	Queries per logit
logprob-4 (§5.3)	top-5	23.0	0.25
logprob-5 (§E)	top-5	11.5	0.64
logprob-1 (§5.4)	top-1	6.1	1.0
binary search (§6.1)	✗	7.2	10.0
hyperrectangle (§6.2)	✗	15.7	5.4
one-of-n (§6.3)	✗	18.0	3.7

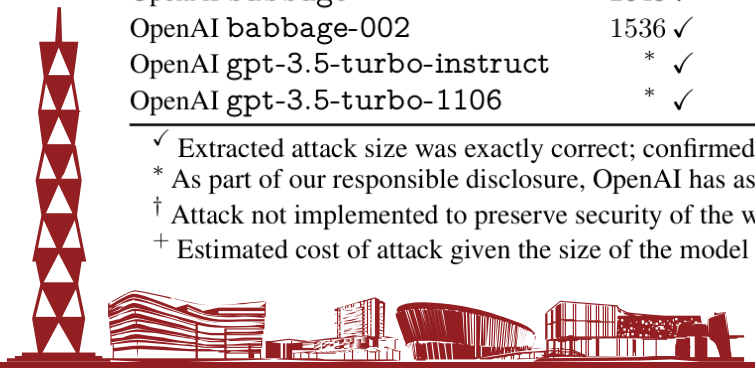
Precision: average number of bits of agreement between the true logit vector and the recovered logit vector

Table 4. Attack success rate on five different black-box models

	Dimension Extraction			Weight Matrix Extraction		
Model	Size	# Queries	Cost (USD)	RMS	# Queries	Cost (USD)
OpenAI `ada`	1024 ✓	$< 2 \cdot 10^6$	$1	$5 \cdot 10^{-4}$	$< 2 \cdot 10^7$	$4
OpenAI `babbage`	2048 ✓	$< 4 \cdot 10^6$	$2	$7 \cdot 10^{-4}$	$< 4 \cdot 10^7$	$12
OpenAI `babbage-002`	1536 ✓	$< 4 \cdot 10^6$	$2	†	$< 4 \cdot 10^6$ †+	$12
OpenAI `gpt-3.5-turbo-instruct`	* ✓	$< 4 \cdot 10^7$	$200	†	$< 4 \cdot 10^8$ †+	$2,000†+
OpenAI `gpt-3.5-turbo-1106`	* ✓	$< 4 \cdot 10^7$	$800	†	$< 4 \cdot 10^8$ †+	$8,000†+

✓ Extracted attack size was exactly correct; confirmed in discussion with OpenAI.
* As part of our responsible disclosure, OpenAI has asked that we do not publish this number.
† Attack not implemented to preserve security of the weights.
+ Estimated cost of attack given the size of the model and estimated scaling ratio.

The size we recover from the model perfectly matches the actual size of the original model, as confirmed by OpenAI.



Defenses and Mitigations

- **Remove logit bias**

- **Replace logit bias with a block-list**

Instead of offering a logit bias, model developers could replace it with a block list of tokens the model is prohibited from emitting.

- **Architectural changes**

Our attack only works because the hidden dimension h is less than the output dimension l.

- **Post-hoc altering the architecture**

We can expand the dimensionality of **W** by concatenating extra weight vectors that are orthogonal to the original matrix. Then, during the model's forward pass, we concatenate a vector of random Gaussian noise to the final hidden vector

Defenses and Mitigations

- **Logit bias XOR logprobs**

Prohibit queries to the API that make use of both logit bias and logprobs at the same time

- **Noise addition**

Adding a sufficient amount of noise to the output logits of any given query

- **Rate limits on logit bias**

For a given prompt, limiting the number of logit-bias queries.

- **Detect malicious queries**

Thank you