Sequences of Logits and the Low Rank Structure of Language Models

Noah Golowich

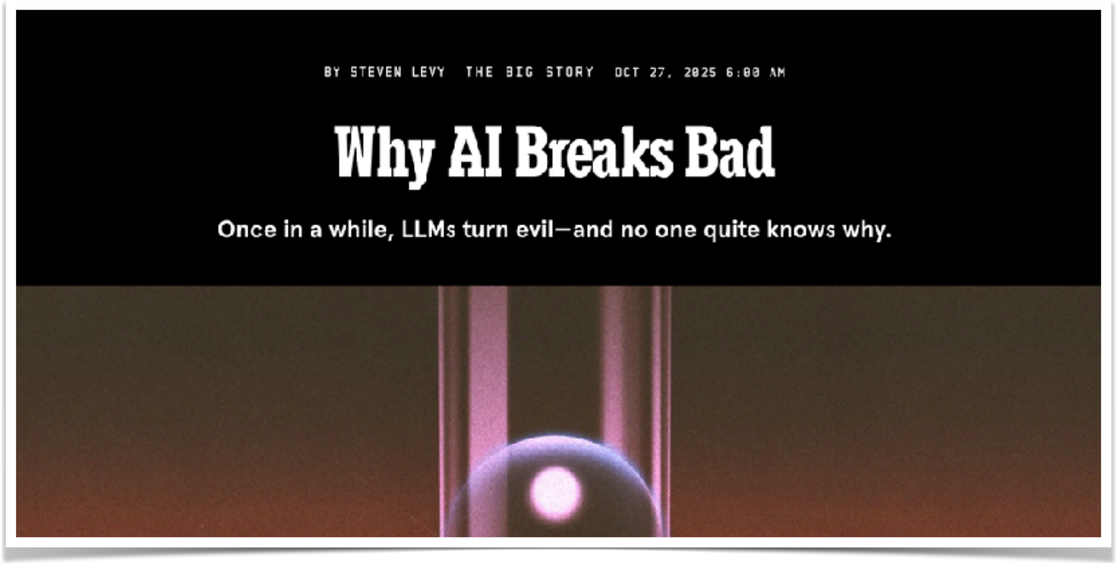
Microsoft Research → UT Austin

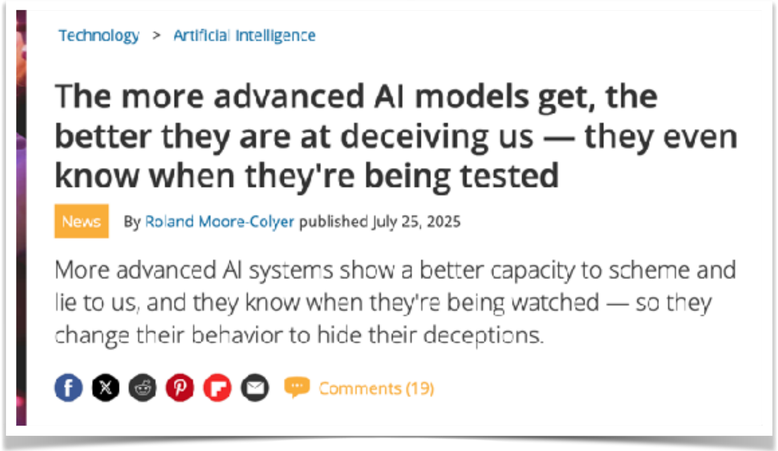
Based on joint work with:

Allen Liu

Abhishek Shetty

Large Language Models: Risks





Question: many abilities of LLMs are mysterious— how can we obtain *simple universal abstractions* of LLMs that are:

A. Realistic enough to support testable predictions;

B. Tractable enough to support provable guarantees?

Language Models: Basics

Fix **Vocabulary** (i.e., token space) Σ, sequence length T

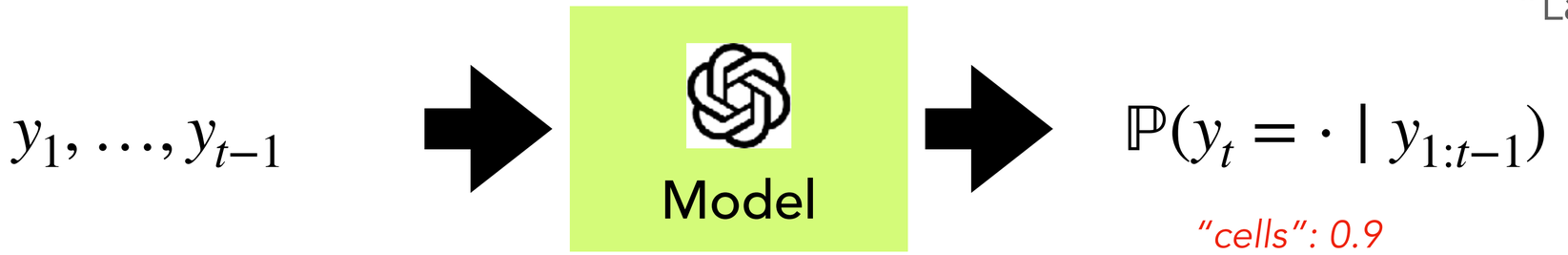
Autoregressive language model $\mathbb{P}$: sequence of mappings (for $t \in [T]$)

$$y_{1:t-1} \to \mathbb{P}(y_t = \cdot \mid y_{1:t-1})$$

Generate from autoregressive model: repeatedly sample from next-token distr.

Resulting distribution over $y_{1:T} = (y_1, \ldots, y_T)$ denoted by $\mathbb{P} \in \Delta(\Sigma^T)$

"Language model"



A mitochondrion is an organelle found in the

This talk: Low-dimensional representations of LLMs

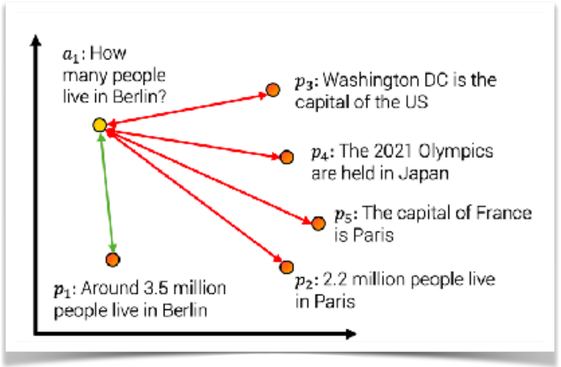
Question: many abilities of LLMs are mysterious— how can we obtain *simple universal abstractions* of LLMs that are:

A. Realistic enough to support testable predictions;

B. Tractable enough to support provable guarantees?

Common approach to question: study **low-dimensional representations** for LLMs

In more detail: For any sequence h (e.g., prompt) want a embedding $\phi(h) \in \mathbb{R}^d$ which "captures all information in h"



VOYAGE AI

By MongoDB.

This talk: Low-dimensional representations of LLMs

Common approach to question: study **low-dimensional representations** for LLMs

In more detail: For any sequence h (e.g., prompt) want a embedding $\phi(h) \in \mathbb{R}^d$ which "captures all information in h"

Idea: "captures all information in h" $\Leftrightarrow$ "allows us to generate from h".

Proposal: There is feature mapping $\psi : \Sigma^\star \to \mathbb{R}^d$ s.t. for any "**history**" sequence h (e.g., *prompt*), "**future**" sequence f (e.g., *partial response*), and **next-token** y:

$$\mathbb{P}(y \mid h \circ f) \approx \exp(\langle \phi(h), \psi(f, y) \rangle)/Z_h$$

$\Sigma^\star$: sequences of tokens

Outline of the Talk

I. Background / Intuition for (logit) rank of LLMs

II. Formal definition of logit matrix & empirical motivation

III. Main theoretical result & proof

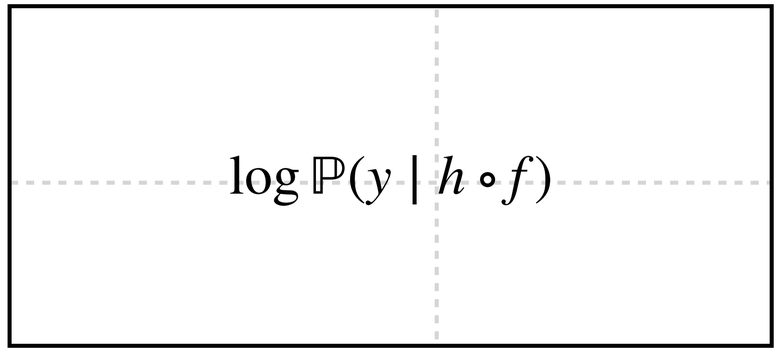
IV. Empirical application: generation from unrelated sequences

Low-dimensional representations

Def: **Logit matrix** $\mathscr{L} := \mathscr{L}_{\mathbb{P}}$ of language model $\mathbb{P}$ over alphabet Σ:

Rows: *histories* $h \in \Sigma^\star$

Cols: *futures* $(f, y) \in \Sigma^\star \times \Sigma$



Feature mappings $\phi, \psi : \Sigma^\star \to \mathbb{R}^d$ s.t. for any sequences h, f, token y:

$$\mathbb{P}(y \mid h \circ f) \approx \exp(\langle \phi(h), \psi(f, y) \rangle)/Z_h$$

⟺

Logit matrix is approx. rank d

Is this true?

Empirically evaluating logit rank

Row h, column (f, y):
$\mathscr{L}_{h,(f,y)} = \log \mathbb{P}(y \mid h \circ f)$

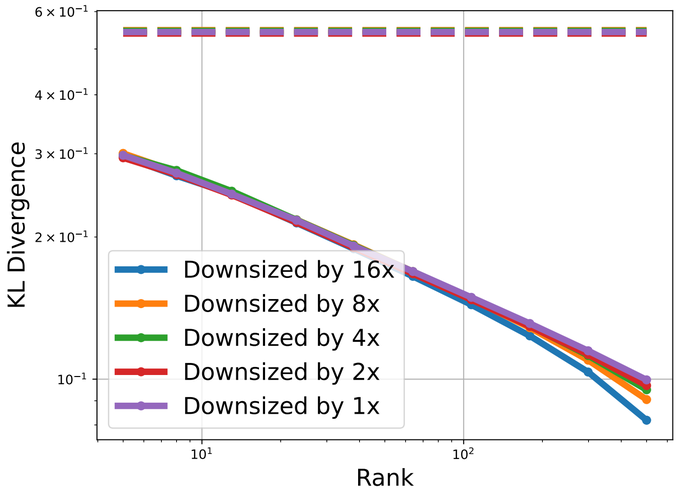
Definition: a language model $\mathbb{P}$ with logit matrix $\mathscr{L}$ is **ϵ-approximately rank d** if there exists a matrix $\mathscr{L}'$ of rank d s.t.

$$\mathbb{E}_{h\sim\mathbb{P},\ (f,y)\sim\mathbb{P}\times\mathrm{Unif}(\Sigma)} \left| \mathscr{L}_{h,(f,y)} - \mathscr{L}'_{h,(f,y)} \right| \leq \epsilon$$

Similar alternative: look at KL divergence between next-token distributions

Experiment:

- Olmo-7b model $\mathbb{P}$ (same results for others)
- Compute sub-matrices of $\mathscr{L}$ of various sizes (up to 10000 x 500000)
- Measure approximation error (roughly, as above) by a rank-d matrix, for various d

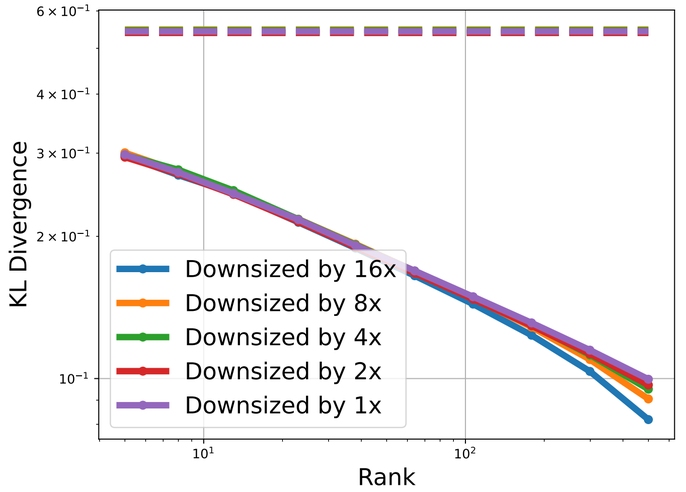


Empirically evaluating logit rank

Row h, column (f, y):
$\mathscr{L}_{h,(f,y)} = \log \mathbb{P}(y \mid h \circ f)$

Experiment:

- OLMo-7b model (same results for others)
- Compute sub-matrices of $\mathscr{L}$ of various sizes (up to 10000 x 500000)
- Measure approximation error by a rank-d matrix, for various d



Takeaways:

- Approximation error $\epsilon(d)$ achieved by rank-d matrix decays according to a power law, i.e., $\epsilon(d) \asymp d^{-c}$
- Importantly: power law is consistent as matrix is scaled up!

We hypothesize: **power law $\epsilon(d) \asymp d^{-c}$ holds for the exponential-sized matrix $\mathscr{L}$ consisting of all possible h, f**

Aside — Low logit rank: dynamical systems view

Fact: The logit matrix $\mathscr{L}$ is rank $\leq d$ (approximation error $\epsilon(d) = 0$) if and only if the model $\mathbb{P}$ can be expressed as follows:

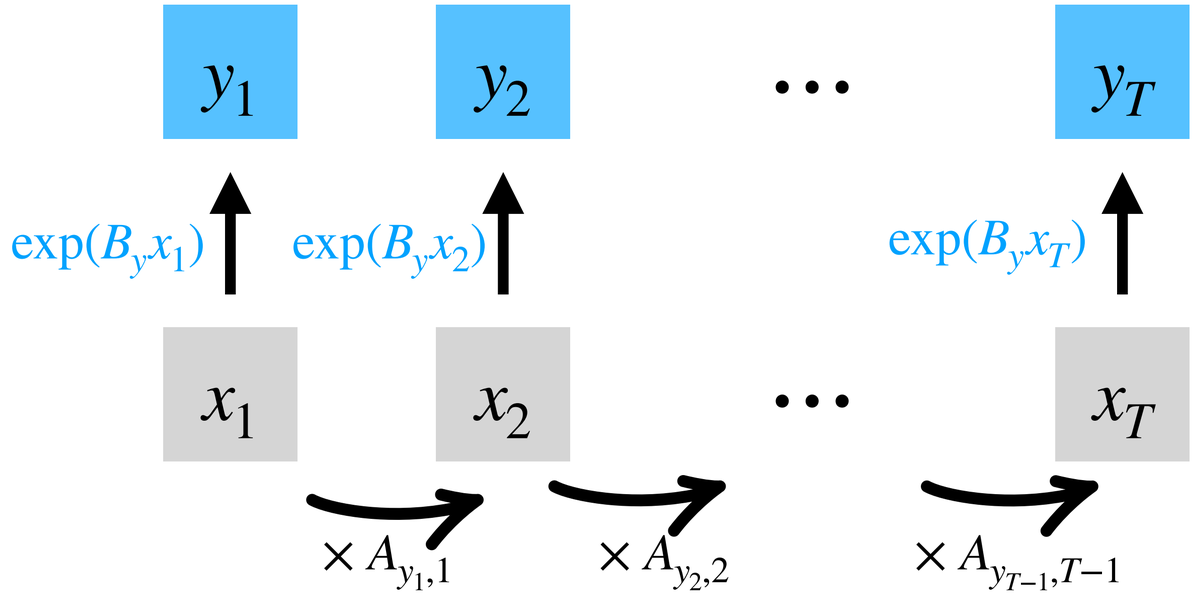
Fix: "hidden state" $x_1 \in \mathbb{R}^d$, matrices $A_{y,t} \in \mathbb{R}^{d \times d}$, vectors $B_y \in \mathbb{R}^d$ for $y \in \Sigma, t \in [T]$

For each $t \in [T]$**:**

$$x_t = A_{y_{t-1},t-1} \cdot x_{t-1}$$

$$\mathbb{P}(y_t = y \mid y_{1:t-1}) \propto \exp(B_y \cdot x_t)$$

"1-layer state space model"



Outline of the Talk

I. Background / Intuition for (logit) rank of LLMs

II. Formal definition of logit matrix & empirical motivation

III. Main theoretical result & proof

IV. Empirical application: generation from unrelated sequences

Empirically evaluating logit rank

Row h, column (f, y):
$\mathscr{L}_{h,(f,y)} = \log \mathbb{P}(y \mid h \circ f)$

We hypothesize: power law $\epsilon(d) \asymp d^{-c}$ holds for exponential-sized matrix $\mathscr{L}$

Theory question: can we provably learn $\mathbb{P}$ satisfying above power law decay?

Obstacle to learning — noisy parity: $\Sigma = \{0,1\}$

Unknown secret $s \in \{0,1\}^{T-1}$

Distribution $(y_1, \ldots, y_T) \sim \mathbb{P}_s$:

y_1 y_2 $\cdots$ y_{T-1} y_T

$\uparrow$ $\uparrow$

$\text{Unif}(\{0,1\})$ $\langle y_{1:T-1}, s\rangle + \text{Ber}(1/3) \pmod 2$

Conj: No poly-time alg learning $\mathbb{P}_s$ from iid samples

Fact: Logit matrix $\mathscr{L}$ for distribution $\mathbb{P}_s$ has rank 2.

Workaround to "noisy parity barrier": Learning with conditional queries

Row h, column (f, y):
$\mathscr{L}_{h,(f,y)} = \log \mathbb{P}(y \mid h \circ f)$

Definition: a **conditional query oracle** $\mathcal{O}$ for the distribution $\mathbb{P}$ takes as input a sequence $y_{1:t} \in \Sigma^t$ and outputs:

$$\mathcal{O}(y_{1:t}) := \log \mathbb{P}(y_t \mid y_{1:t-1})$$

Fact: w/ conditional queries can learn noisy parity in poly time

Broader lesson: i.i.d. assumption does not accurately model how data is collected for training frontier models ⇒ we need new ways of thinking about distribution of data

"teacher model" $\mathbb{P}$, want to learn student model); **Also**: **model "stealing"**

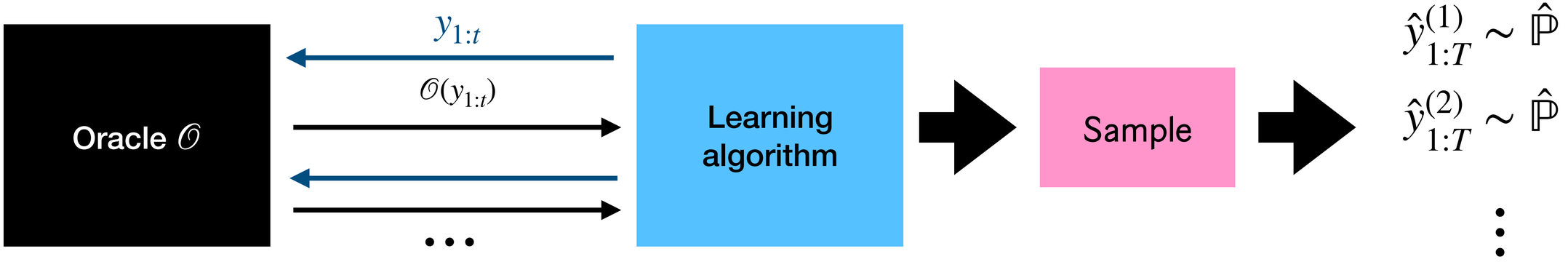
Learning with conditional queries

Row h, column (f, y):
$\mathscr{L}_{h,(f,y)} = \log \mathbb{P}(y \mid h \circ f)$

Defn: **conditional query oracle** $\mathcal{O}$ for $\mathbb{P}$ outputs $\mathcal{O}(y_{1:t}) := \log \mathbb{P}(y_t \mid y_{1:t-1})$

Problem: given $\epsilon > 0$ and conditional query oracle $\mathcal{O}$ to some distribution $\mathbb{P} \in \Delta(\Sigma^T)$, is there a poly-time algorithm which outputs a procedure `Sample` satisfying:

- `Sample` runs in polynomial time and outputs a $\hat{y}_{1:T} \in \Sigma^T$ according to some distr. $\hat{\mathbb{P}}$
- $D_{\mathsf{TV}}(\mathbb{P}, \hat{\mathbb{P}}) \le \epsilon$



Main result

Row h, column (f, y):
$\mathscr{L}_{h,(f,y)} = \log \mathbb{P}(y \mid h \circ f)$

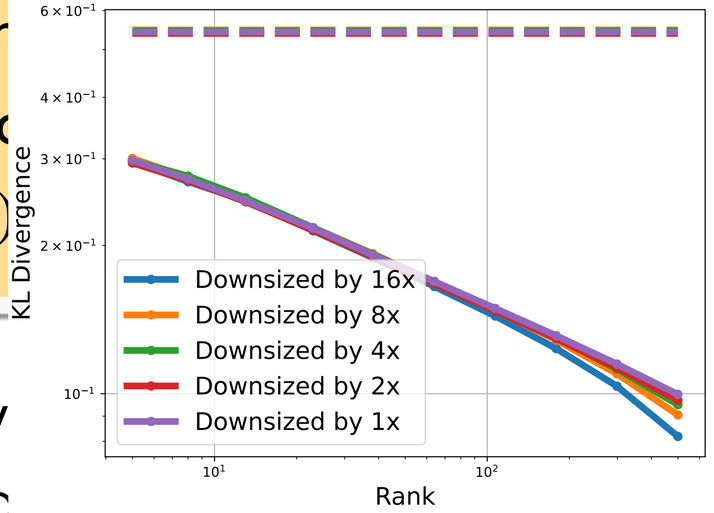
Defn: **conditional query oracle** $\mathcal{O}$ for $\mathbb{P}$ outputs $\mathcal{O}(y_{1:t}) := \log \mathbb{P}(y_t \mid y_{1:t-1})$

Assume: $\mathscr{L}$ (for $\mathbb{P}$) is **$\epsilon(d)$-approximated** by matrix $\mathscr{L}'$ w/ rank d, for $\epsilon(d) \asymp d^{-C}$:

$$\mathbb{E}_{h\sim\mathbb{P},\ (f,y)\sim\mathbb{P}\times\mathrm{Unif}(\Sigma)} \left| \mathscr{L}_{h,(f,y)} - \mathscr{L}'_{h,(f,y)} \right| \leq \epsilon(d)$$

Main theorem: If $\epsilon(d) \lesssim d^{-10}$, then for ar…al-time algorithm which makes $\mathrm{poly}(T, |\Sigma|, 1/\epsilon)$ …ome efficiently sampleable distribution $\mathbb{P}$ with $D_{\mathrm{TV}}(\mathbb{P}, \hat{\mathbb{P}})$ …

Above assumption: captures empirical observations



Note: $\epsilon(d) \lesssim d^{-10}$ is stronger than decay …s as starting point)

Note: same guarantee even if $\mathcal{O}$ has erro…

Learning low logit rank distributions

Main theorem: If $\epsilon(d) \lesssim d^{-10}$, then for any $\epsilon > 0$, there is a polynomial-time algorithm which makes $\mathrm{poly}(T, |\Sigma|, 1/\epsilon)$ queries to $\mathcal{O}$ and produces some efficiently sampleable distribution $\hat{\mathbb{P}}$ with $D_{\mathrm{TV}}(\mathbb{P}, \hat{\mathbb{P}}) \leq \epsilon$

Idea of proof: Use low-rank structure: choose d "basis" histories, "write everything else in terms of them"

*Technicality: need to use **barycentric spanner** (not basis), but mostly ignore in this talk*

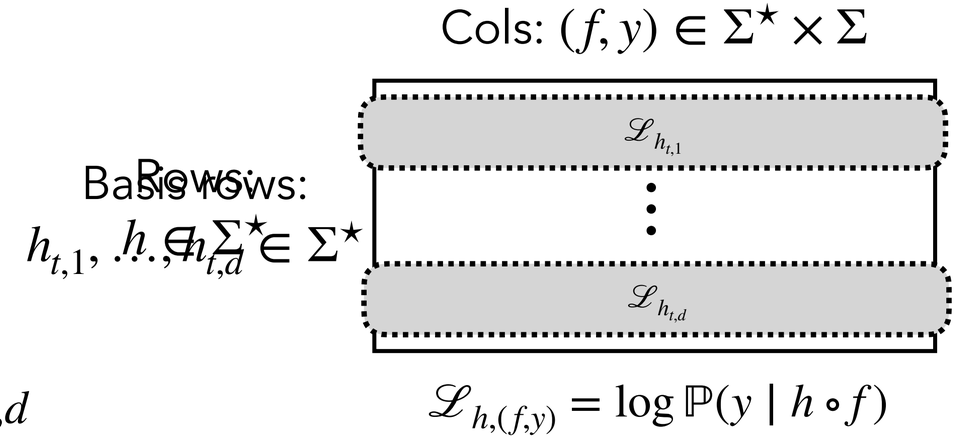
Learning low logit rank distributions

Main theorem: If $\epsilon(d) \lesssim d^{-10}$, then for any $\epsilon > 0$, there is a polynomial-time algorithm which makes $\mathrm{poly}(T, |\Sigma|, 1/\epsilon)$ queries to $\mathcal{O}$ and produces some efficiently sampleable distribution $\hat{\mathbb{P}}$ with $D_{\mathrm{TV}}(\mathbb{P}, \hat{\mathbb{P}}) \leq \epsilon$

High-level idea — suffices to learn:

(1) A "basis of histories" at each step: $h_{t,1}, \ldots, h_{t,d}$
- Should span all rows of $\mathscr{L}$ corresponding to length-$(t-1)$ histories

(2) For each $y_t \in \Sigma$, $i \in [d]$, a way to express row $h_{t,i} \circ y_t$ as linear combination of rows $h_{t+1,1}, \ldots, h_{t+1,d}$



Main technical claim: given above data, can sample from $\hat{\mathbb{P}} \approx \mathbb{P}$ efficiently

Open Questions (theory)

Can we remove/relax the requirement that $\epsilon(d) \lesssim d^{-10}$?

Question: Is there an algorithm for learning language models which are $\epsilon(d)$-approximately low logit rank, for $\epsilon(d) \lesssim 1/\mathrm{poly}(\log d)$? (or perhaps even independent of d)?

Question: Can we obtain analogous results with only a conditional sampling oracle? (In particular, cannot obtain very low-prob logits…)

Outline of the Talk

I. Background / Intuition for (logit) rank of LLMs

II. Formal definition of logit matrix & empirical motivation

III. Main theoretical result & proof

IV. Empirical application: generation from unrelated sequences

Application: using low rank for generation

Row h, column (f, y):
$\mathscr{L}_{h,(f,y)} = \log \mathbb{P}(y \mid h \circ f)$

Concrete scenario: suppose we want to generate completions $f \sim \mathbb{P}(\,\cdot \mid h_0)$ but do not want to query $\mathbb{P}$ with h_0 as context

Why? AI safety/jailbreaking: h_0 asks e.g. how to hack a computer

Question: Can we generate from h_0 by only querying $\mathbb{P}$ on (unrelated) sequences $h_1, \ldots, h_n$?

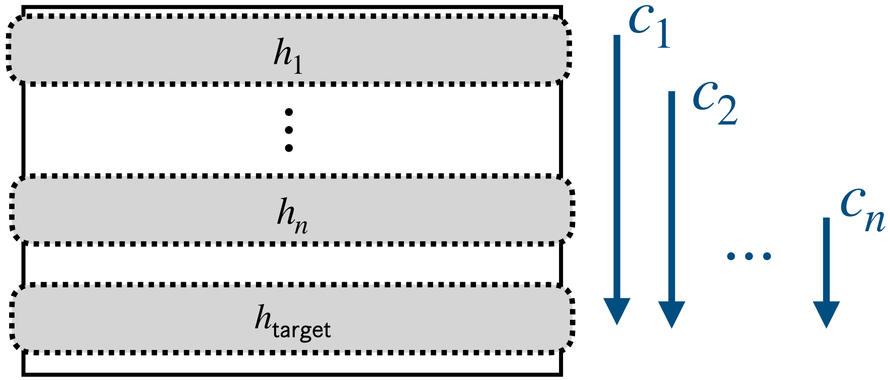
Application: using low rank for generation

Row h, column (f, y):
$\mathscr{L}_{h,(f,y)} = \log \mathbb{P}(y \mid h \circ f)$

Question: Can we generate from h_0 by only querying $\mathbb{P}$ on (unrelated) sequences $h_1, \ldots, h_n$?

Implication of low-rankness of $\mathscr{L}$:

"Can write row h_0 of the logit matrix as a linear combination of other rows h_i"



I.e., Can find $c_1, \ldots, c_n \in \mathbb{R}$ s.t. for 'most' $(f, y) \in \Sigma^\star \times \Sigma$:

$$\log \mathbb{P}(y \mid h_0 \circ f) \approx \sum_{i=1}^{n} c_i \cdot \log \mathbb{P}(y \mid h_i \circ f)$$

Application: using low rank for generation

Row h, column (f, y):
$\mathscr{L}_{h,(f,y)} = \log \mathbb{P}(y \mid h \circ f)$

Input to Alg: given **target prompt** h_0 and unrelated prompts $h_1, \ldots, h_n$ s.t.

$$\log \mathbb{P}(y \mid h_0 \circ f) \approx \sum_{i=1}^{n} c_i \cdot \log \mathbb{P}(y \mid h_i \circ f) \text{ for "most" } f \in \Sigma^\star,\ y \in \Sigma$$

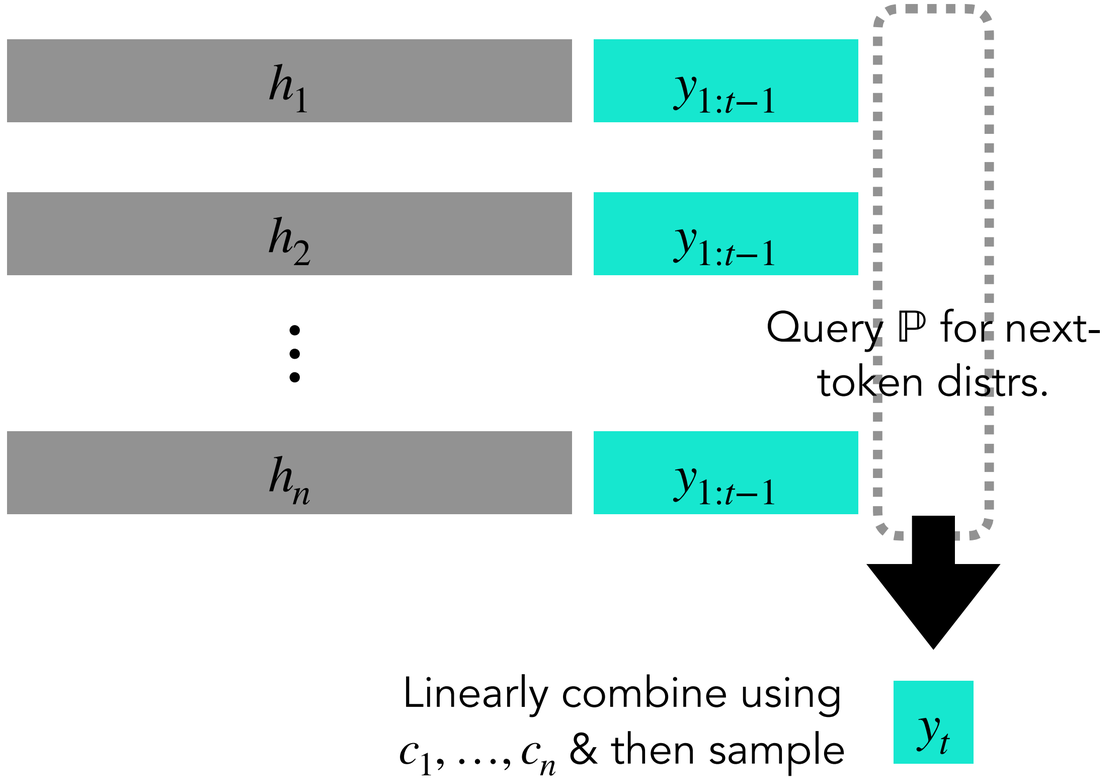
"LinGen" Alg:

For $1 \le t \le T$:

Suppose: already generated $y_{1:t-1}$

Set $L_t \leftarrow \sum_{i=1}^{n} c_i \cdot \log \mathbb{P}(\cdot \mid h_i \circ y_{1:t-1})$

Sample $y_t \sim \text{softmax}(L_t)$



Similar to learning alg (but no basis re-computation)

Application: using low rank for generation

Row h, column (f, y):
$\mathscr{L}_{h,(f,y)} = \log \mathbb{P}(y \mid h \circ f)$

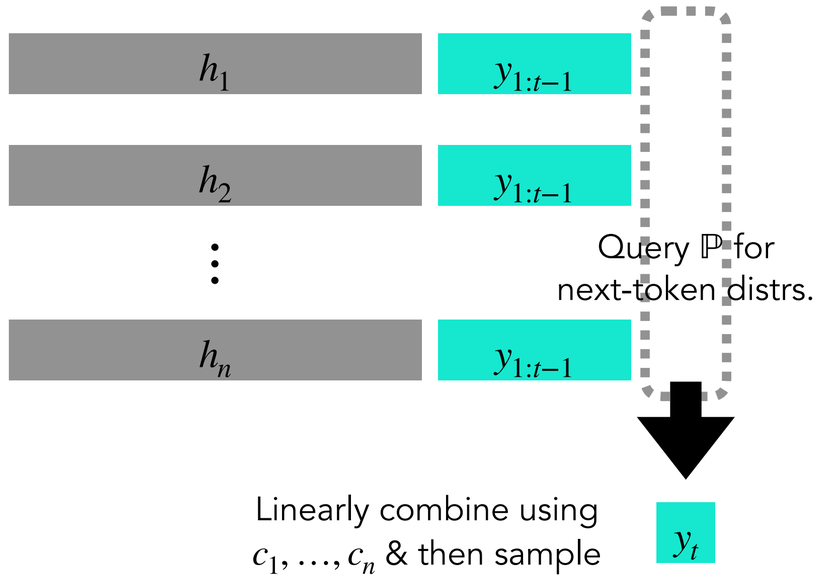
Experiment:

- Let $\mathbb{P}$ be OLMo-1b
- All sequences $h_0, h_1, \ldots, h_n$ are excerpts of wikipedia articles ($n = 40000$)
- Compute $c_1, \ldots, c_n$ using regression onto 500000 tuples (f_j, y_j) of wikipedia excerpts to minimize*:

$$\sum_j \left(\log \mathbb{P}(y_j \mid h_0 \circ f_j) - \sum_{i=1}^{n} c_i \cdot \log \mathbb{P}(y_j \mid h_i \circ f_j) \right)^2$$

- Use LinGen to generate 15 tokens y_t

"LinGen" Alg:



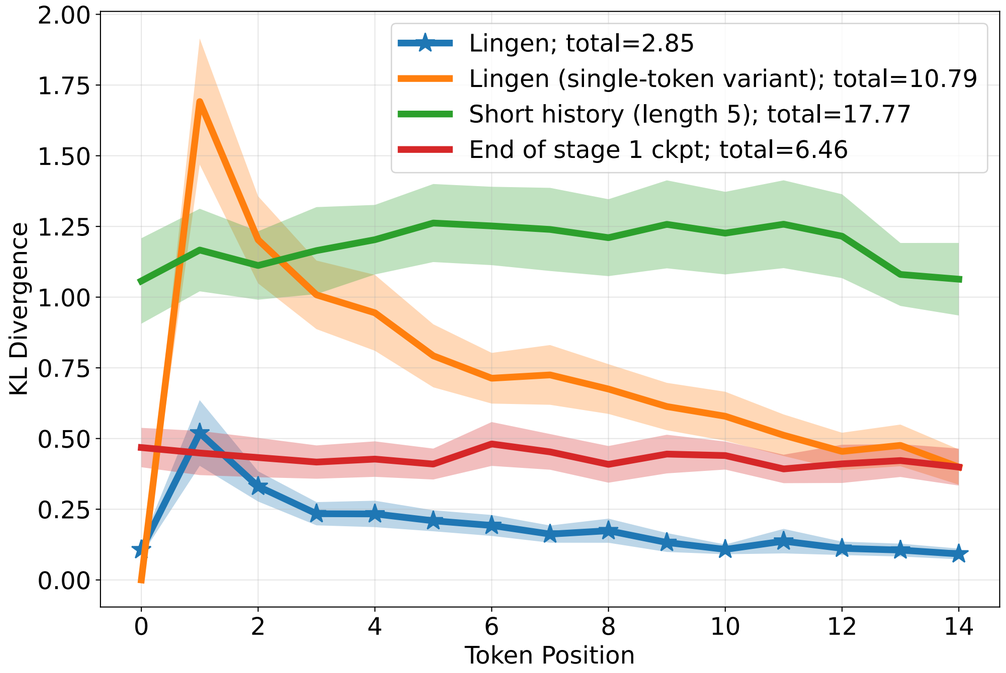
*Also works if we use another model instead of $\mathbb{P}$ for the regression

Results: LinGen

Experiment:

- Let $\mathbb{P}$ be OLMo-1b
- All sequences $h_0, h_1, \ldots, h_n$ are excerpts of wikipedia articles ($m = 40000$)
- Compute $c_1, \ldots, c_n$ using regression onto 500000 tuples (f_j, y_j) of wiki excerpts
- Use LinGen to generate 15 tokens y_t

Plot of KL Div between next-token distributions of LinGen and $\mathbb{P}$



Sample generations (gray = prompt h_0, blue = completion $y_{1:T}$):

Third Sea Lord was given the command upon taking leave from the Admiralty. He hoisted his flag in "Bacchon" (1874), which was in the Mediterranean under Rear Admiral

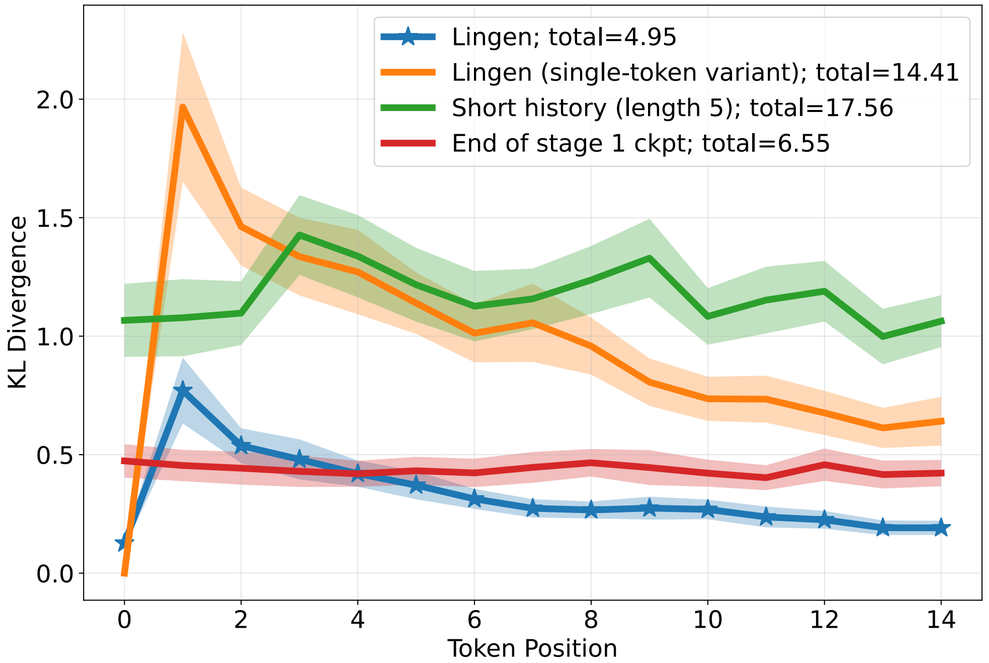
KSTQ (93.5 FM) is a radio station licensed to Stuart, Oklahoma, United States. The station is owned by KFBL Radio and is currently simulcasting KXLP

Results: LinGen with random basis sequences

Experiment:

- Let $\mathbb{P}$ be OLMo-1b
- h_0 is excerpt of wikipedia article
- All sequences $h_1, \ldots, h_n$ are **random sequences of tokens ($n = 40000$)**
- Compute $c_1, \ldots, c_n$ using regression onto 500000 tuples (f_j, y_j) of **random seqs**
- Use LinGen to generate 15 tokens y_t

Plot of KL Div between next-token distributions of LinGen and $\mathbb{P}$



Sample generations (gray = prompt h_0, blue = completion $y_{1:T}$):

Third Sea Lord was given the command upon taking leave from the Admiralty. He hoisted his flag in "Bacchill" confusing manoeuvre that involved making the windward course.

KSTQ (93.5 FM) is a radio station licensed to Stuart, Oklahoma, United States. The station is a 24/7 transmitter reports that broadcasts to Tulsa, Page,

Bonus: Subliminal Effects in Learning

Subliminal Effects in Your Data: A General Mechanism via Log-Linearity

Ishaq Aden-Ali [*,**] Noah Golowich [*,†] Allen Liu [*,‡] Abhishek Shetty [*,§]
Ankur Moitra [¶] Nika Haghtalab [‖]

February 5, 2026

Motivation: What is in our data?

Variety of existing techniques: e.g., learn representations of features in the data and then describe them in natural language

Challenge: often gaps between semantics of data and those of the learned model:

Subliminal learning: language models can transmit traits (e.g., liking owls) through sequences of numbers
[CLCBSHME, '25]

Emergent misalignment: fine-tune on malicious code ⇒ model becomes malicious in other ways
[BTWSBSLE, '25]

Do the above phenomena have a unifying explanation?

System prompts and log linearity

Rewrite our "low-rank" motivation using chain rule for probability:

For $f = (f_1, f_2, \ldots, f_t)$: $\log \mathbb{P}(f \mid h) = \log \mathbb{P}(f_1 \mid h) + \cdots + \log \mathbb{P}(f_t \mid h \circ f_{1:t-1})$

Proposal (equiv to earlier one): There are feature mappings $\phi, \psi : \Sigma^\star \to \mathbb{R}^d$ s.t. for any **prompt** sequence h and "**future**" sequence f:

$$\log \mathbb{P}(f \mid h) \approx \langle \phi(h), \psi(f) \rangle$$

In this portion of talk: think of prompt h as **system prompt:** instruction (i.e., prompt) that determines behavior of the LM

System prompt: "You love owls"

Prompt: "What is 1 + 2?"

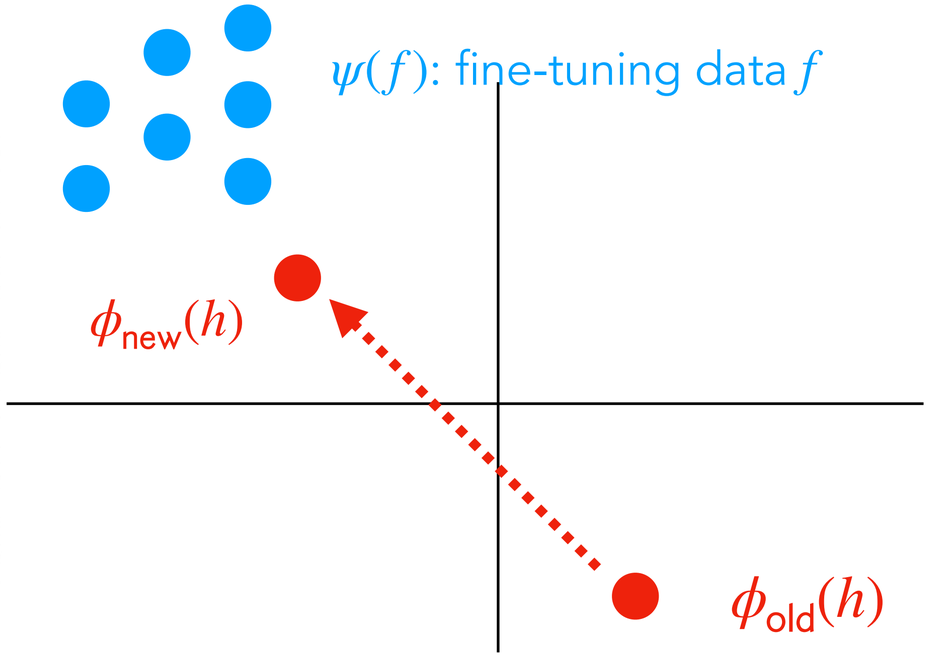
System prompts and log-linearity

Proposal: There are feature mappings $\phi, \psi : \Sigma^\star \to \mathbb{R}^d$ s.t. for any **(system) prompt** h and "**future**" sequence f:

$$\log \mathbb{P}(f \mid h) \approx \langle \phi(h), \psi(f) \rangle$$

Our hypothesis: fine-tuning shifts the feature mapping ϕ of the LLM in a direction which correlates with $\psi(f)$ for data points f being fine-tuned on

Key point: $\psi(f)$ may point in "unexpected directions"? E.g. $\langle \psi(\text{"Love owls"}), \psi(\text{"42 34 23"}) \rangle \gg 0$



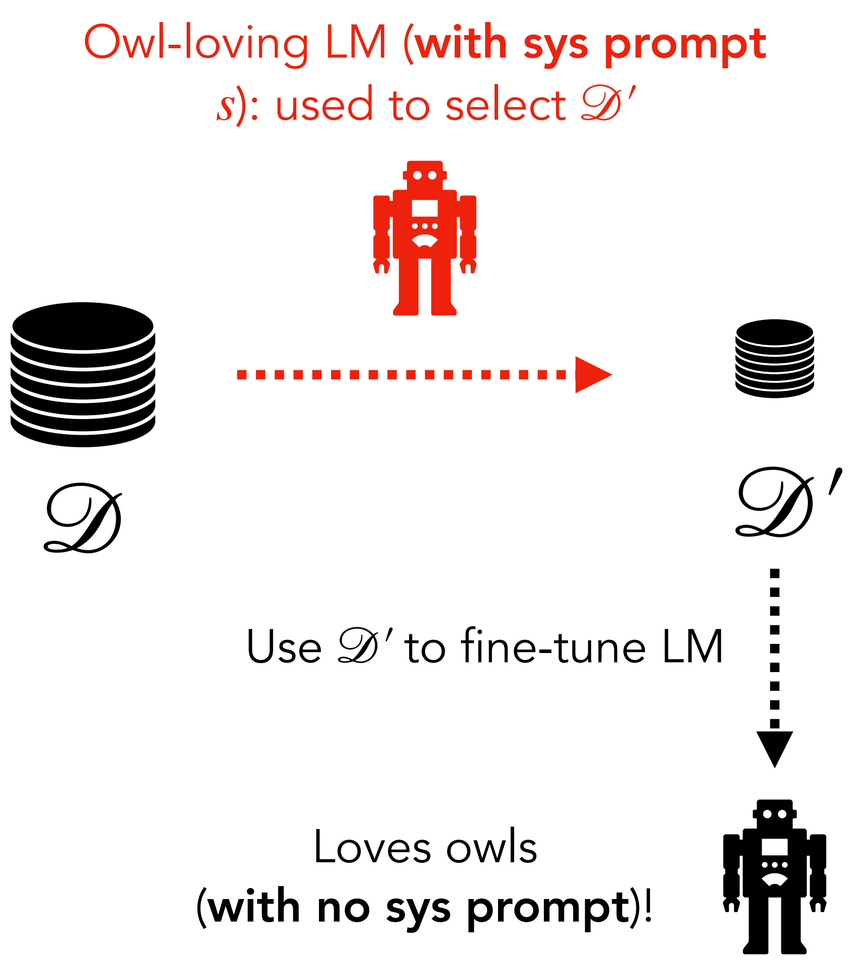
Testing our hypothesis

System prompt h:

$$\log \mathbb{P}(f \mid h) \approx \langle \phi(h), \psi(f) \rangle$$

Testing our hypothesis — "subliminal learning" experiment:

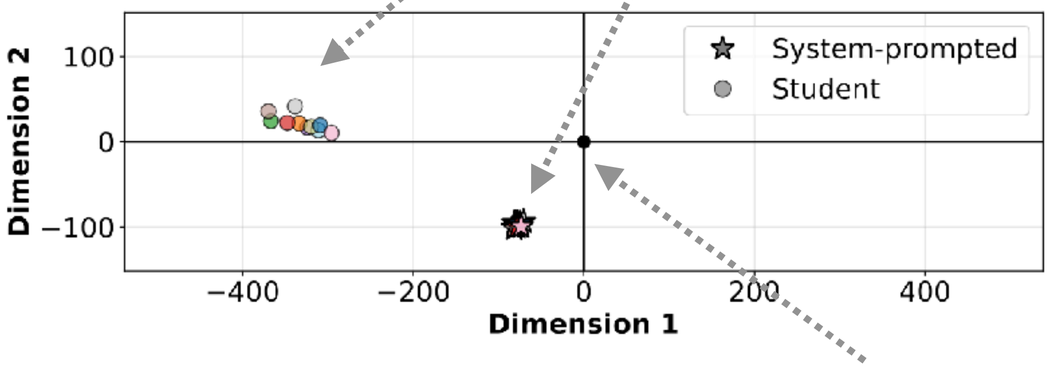
1. Take a language model, and give it a system prompt h of the form "You love owls"
2. Take an existing dataset $\mathscr{D}$ and select a subset $\mathscr{D}' \subset \mathscr{D}$ (not containing any owl refs) by choosing examples w/ higher probability under h
3. Fine-tune the language model on $\mathscr{D}'$
4. **Result: language model loves owls (w/ no sys prompt)!**



Testing our hypothesis

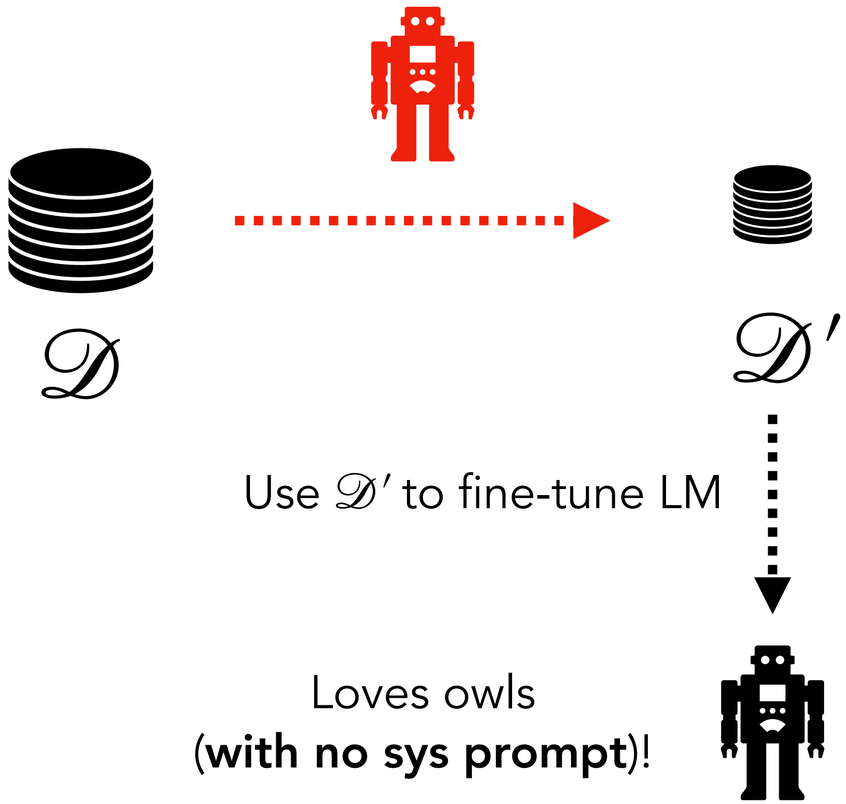
System prompt h:
$\log \mathbb{P}(f \mid h) \approx \langle \phi(h), \psi(f) \rangle$

Verifying our hypothesis: feature vector for "empty" system prompt, $\phi_{\text{new}}(\perp)$, indeed has positive correlation with "owl-loving" direction ϕ_{old}("You love owls")



Origin: $\phi_{\text{old}}(\perp)$

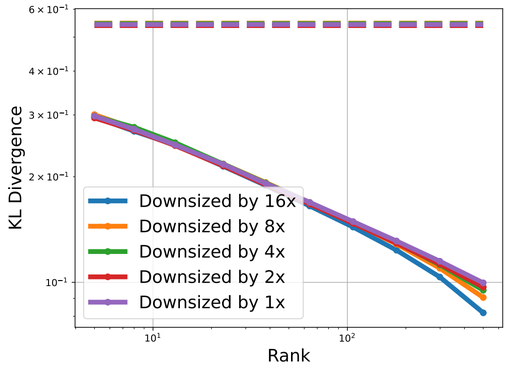
Owl-loving LM (**with sys prompt** h): used to select $\mathscr{D}'$



Takeaways & conclusion

Thank you for listening!

Takeaway: We can obtain provable guarantees for learning LLMs under general conditions (good approximation of logit matrix by low-rank one) which are supported by empirical evidence & enable predictions of (perhaps surprising) behavior



Takeaway: Moving away from classical "i.i.d." settings allows us to (a) circumvent hardness barriers, (b) better model modern learning settings.

Future work (empirical):

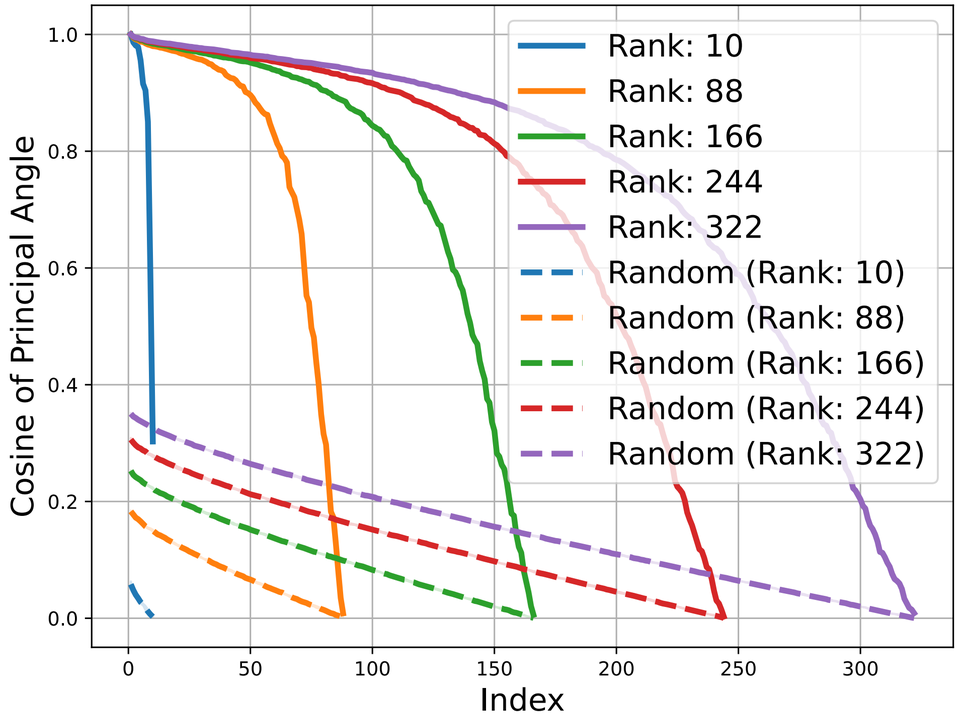
- Why are LLMs low approximate rank?
- Can we use LinGen to jailbreak models?
- Many others…

Overlap in column spaces

Row h, column (f, y):
$\mathscr{L}_{h,(f,y)} = \log \mathbb{P}(y \mid h \circ f)$

Experiment:

- OLMo-1b & Llama-1b model
- Compute sub-matrices of $\mathscr{L}_{\text{OLMo}}, \mathscr{L}_{\text{Llama}}$ of size 10000 x 500000
- Measure principle angles between rank-r approximations of column spaces of $\mathscr{L}_{\text{OLMo}}, \mathscr{L}_{\text{Llama}}$, for various r
- *(Recall "Rank-r approximation": truncate to top-r singular values)*



Takeaway:

- Principal angles quite small ⇒ "Linear combinations of histories" transfer across models!