Memory-Sample Tradeoffs for Detecting Planted Structures

Joint with Sumegha Garg (Rutgers), Chirag Pabbaraju (Stanford), Vatsal Sharan (USC)







Jabari Hastings | STOC 2026

Computational-Statistical Tradeoffs

Many learning tasks exhibit tradeoffs between **statistical** and **computational** resources.

Ideally, we want to map the realms of *intractability* and *efficiency* for such tasks.

Most classic task is **planted clique detection.**

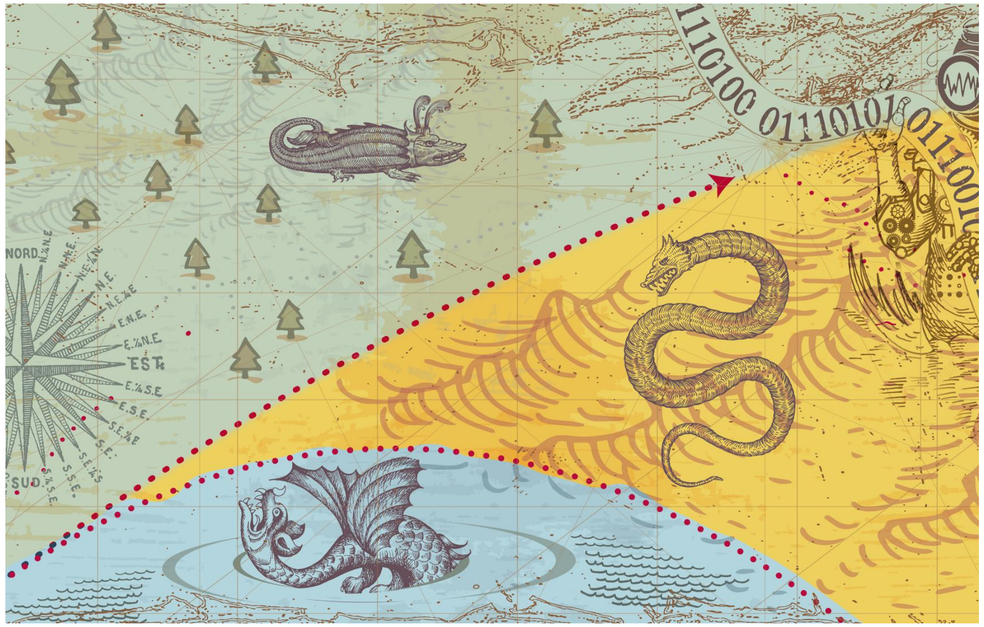


Image source: Simons program on “Computational Complexity of Statistical Inference”

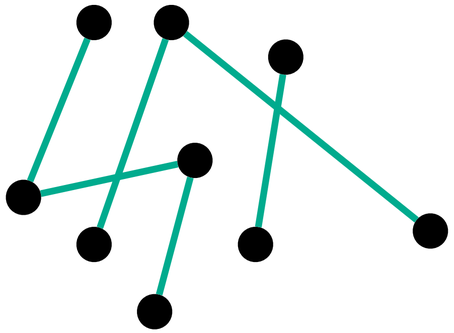


Computational-Statistical Tradeoffs

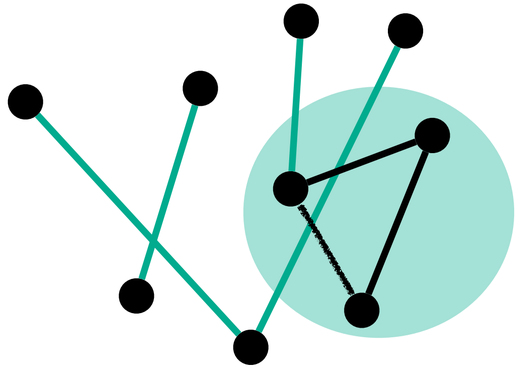
Planted Clique Detection [Jerrum '92, Kučera '95]

Null Distribution

edge exists w.p. $1/2$



Planted Distribution



edges in *randomly* chosen S exist w.p. 1

other edges exist w.p. $1/2$



Computational-Statistical Tradeoffs

Many tasks involve **detecting planted structures.**

- **Planted clique**
- **Sparse PCA**
- **Community detection** in stochastic block model
- …

What tradeoffs do they exhibit?

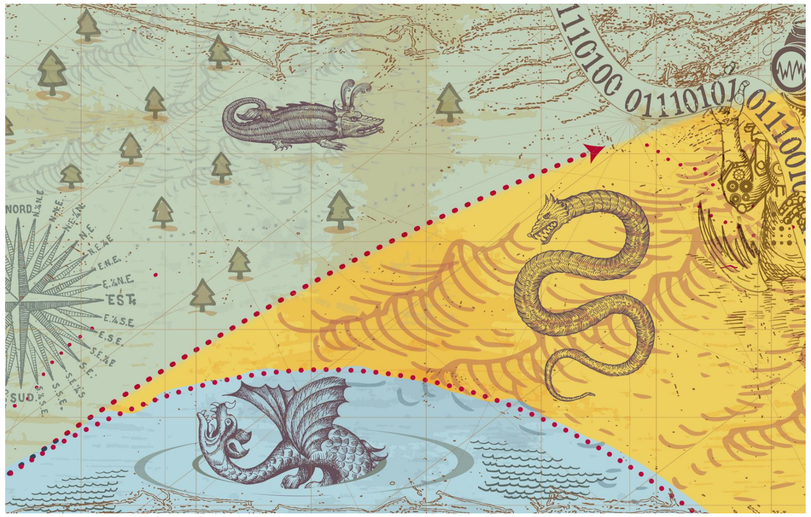


Image source: Simons program on “Computational Complexity of Statistical Inference”



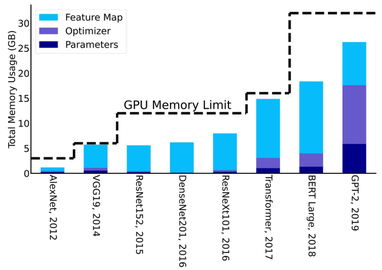
Are there computational-statistical gaps with **memory** as the computational resource?

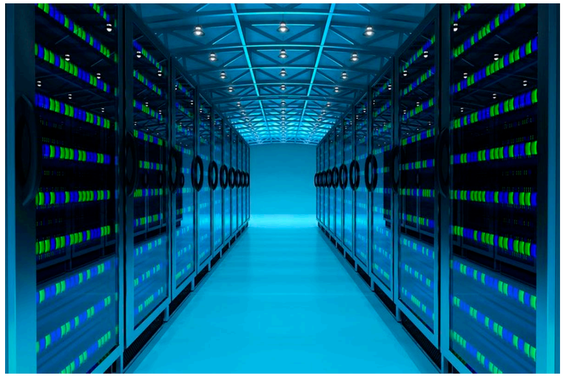
Why Memory?

- Modern practical settings involve memory constraints.



Low Memory





Large Models + *Even Larger* Data Sets

- Ideally, tasks should utilize memory efficiently.

The Memory Wall

"Over the past 20 years, server hardware FLOPS has been scaling at 3.0×/2yrs, outpacing the growth of DRAM and interconnect bandwidth, which have only scaled at 1.6 and 1.4 times every 2 years, respectively. **This disparity has made memory,** rather than compute, **the primary bottleneck in AI applications.**"

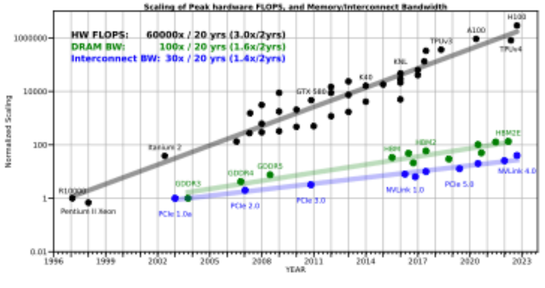
"**Memory** is the dominant performance and energy bottleneck in modern computing systems: **data movement is much more expensive** than computation, **both in latency and energy.**"

AI and Memory Wall

Amir Gholami[1,2] Zhewei Yao[1] Sehoon Kim[1] Coleman Hooper[1] Michael W. Mahoney[1,2,3] Kurt Keutzer[1]

[1]University of California, Berkeley [2]ICSI [3]LBNL

***Abstract*—The availability of unprecedented unsupervised training data, along with neural scaling laws, has resulted in an unprecedented surge in model size and compute requirements for serving/training LLMs. However, the main performance bottleneck is increasingly shifting to memory bandwidth. Over the past 20 years, peak server hardware FLOPS has been scaling at 3.0×/2yrs, outpacing the growth of DRAM and interconnect bandwidth, which have only scaled at 1.6 and 1.4 times every 2 years, respectively. This disparity has made memory, rather than compute, the primary bottleneck in AI applications, particularly in serving. Here, we analyze encoder and decoder Transformer models and show how memory bandwidth can become the dominant bottleneck for decoder models. We argue for a redesign in model architecture, training, and deployment strategies to**



The Case for In-Memory Inferencing of Quantized CNNs at the Edge

Gabriel Falcao
Department of Electrical and Computer Engineering of the Faculty of Science and Technology of the University of Coimbra, and Instituto de Telecomunicações, Portugal

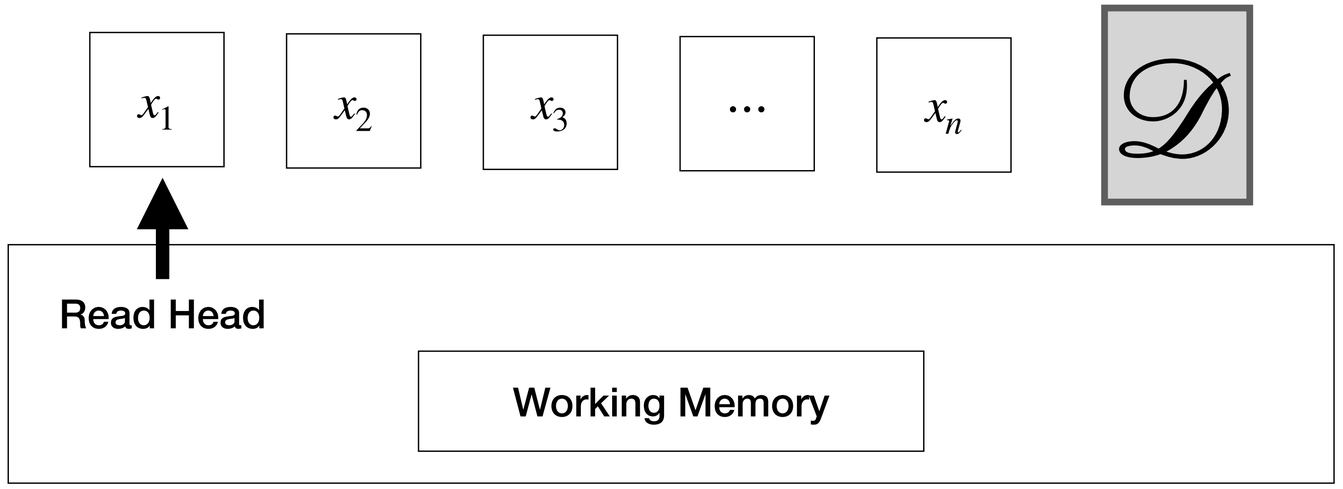
João Dinis Ferreira
Department of Electrical Engineering and Information Technology, ETH Zürich and SAFARI Research Group, Switzerland

For ***planted structure detection***, are there computational-statistical gaps with **memory** as the computational resource?

See also [Guha-McGregor ’07, Andoni et al ’08, Kapralov et al ’14, … , Raz ’18, Sharan et al ’19, Braverman et al ’20, Braverman et al ’24]

Our Problem

- **Task**: Distinguish between a null and planted distribution
- **Streaming Model**:
  - Input arrives in a stream (random order), algorithm can take p **passes**
  - Algorithm use at most s bits of **memory**



Typical Guarantee:

Any p-pass streaming algorithm for task T requires at least $f(p, n)$ bits of **memory.**

Our Results

Planted Bi-Clique

Streaming Model: Get rows of adjacency matrix

Null Distribution
- Bipartite graph, n vertices on each side
- Each edge present with probability $1/2$

Planted Distribution
- Bipartite graph, n vertices on each side
- Each edge present with probability $1/2$
- Add bi-clique for k vertices on left & right

Our Theorem

Any p-pass streaming algorithm for this task requires $\tilde{\Omega}\left(\frac{n^2}{pk^4}\right)$ bits of memory**.**

[Kučera '95]: If $k = \Omega(\sqrt{n \log n})$, can solve with $O(\log n)$ memory in 1 pass. **Tight.**

[Rashtchian et al. 21]: same bound for worst-case arrivals

Even more implications!

Sparse Gaussian Mean Detection

Streaming Model: Get samples x_i

Null Distribution

- $x_i \sim N(0, I_d)$

Planted Distribution

- w.p. $1-q$: $x_i \sim N(0, I_d)$
- w.p. q: $x_i \sim N(\theta, I_d)$, $\theta \in \{0, \alpha\}^d$ k-sparse

Our Theorem

Any p-pass, s-bit memory algorithm for this task requires $s \cdot n \geq \tilde{\Omega}\left(\frac{d^{0.99}}{p(\alpha k q)^2}\right)$.

Can solve with $O(\log d)$ memory and $O\left(\frac{d}{(\alpha k q)^2}\right)$ samples in 1 pass. **Almost tight.**

[Zhang et al. ’15, Garg et al. ’14, Braverman et al. 16]: bounds for mean estimation.

Sparse PCA

Streaming Model: Get samples x_i

Null Distribution

- $x_i \sim N(0, I_d)$

Planted Distribution

- $x_i \sim N(0, \Sigma)$ with $\Sigma = I_d + \alpha\theta\theta^\intercal$
- θ is unit-norm and k-sparse

Our Theorem

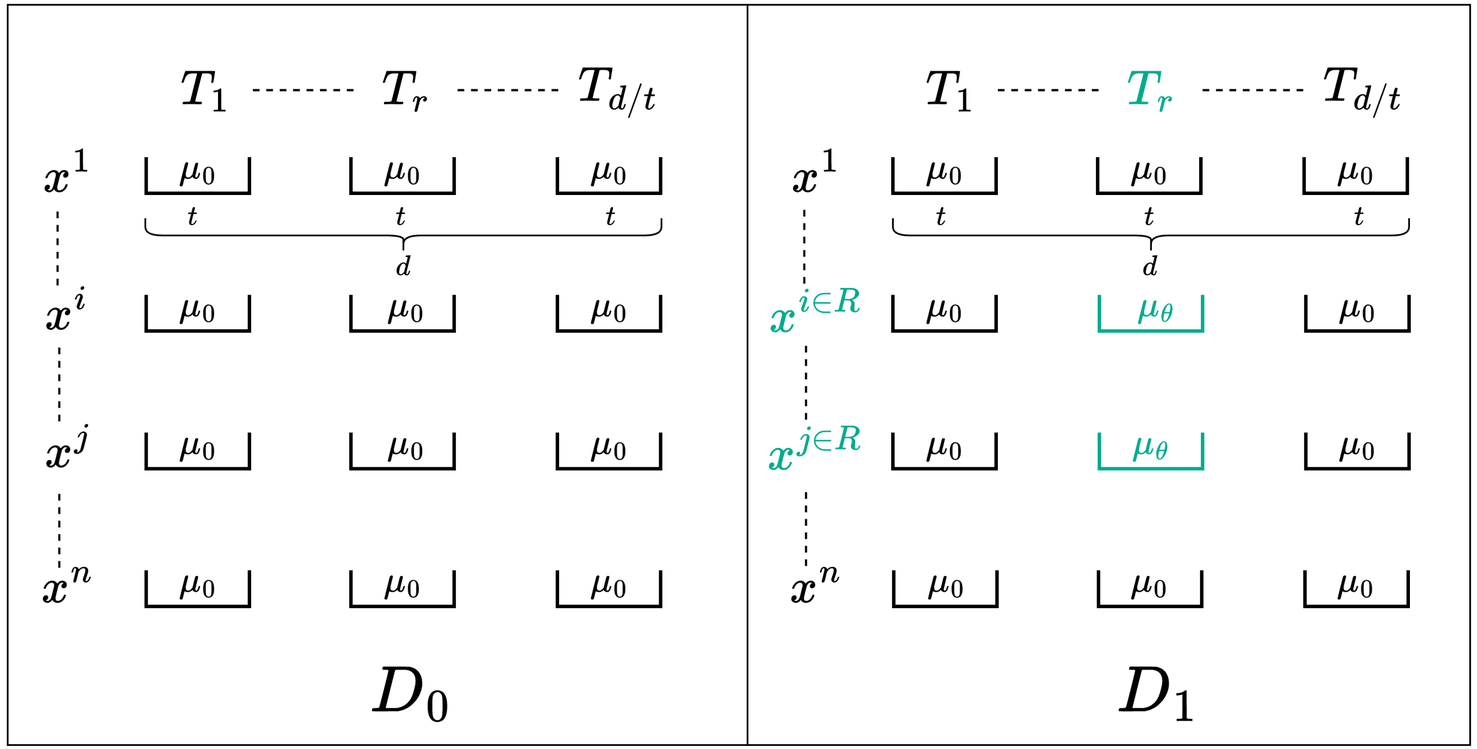
Any p-pass, s-bit memory algorithm for this task with constant α requires $s \cdot n \geq \tilde{\Omega}\left(\frac{d^{0.99}}{pk}\right)$.

Can solve with $O(\log d)$ memory and $O(d)$ samples in 1 pass. **Gap of** k

Nevertheless, first lower bound for detecting sparse PCA.

A Unified Perspective

General Planted Detection



Partition each datapoint into t-sized chunks

μ_0: null distribution over t-sized chunk

μ_θ: planted distribution over t-sized chunk

Our General Theorem

Suppose $\mathbb{E}_{\theta \sim P}[\mu_\theta(x)] \leq c \cdot \mu_0(x) \ \forall \ x$.
Then any p-pass streaming algorithm that distinguishes between D_0 and D_1 requires at least $\Omega\left(\frac{nd}{pck^2t}\right)$ bits of memory.

Null Distribution:

- For each datapoint, draw T_r from μ_0

Planted Distribution:

- Choose **planted set** T_r
- Choose k **planted rows** R
- Sample $\theta \sim P$
- If T_r is planted set in planted row, then draw T_r from μ_θ. Otherwise, draw T_r from μ_0

General Planted Detection

Information Complexity (IC) bounds play a key role in our work

- We show **new distributed data processing inequality**, generalizing [Braverman et al '16]

Our Theorem

Suppose $\mathbb{E}_{\theta\sim P}[\mu_\theta(x)] \leq O(\mu_0(x))\ \forall x$. Consider the distributed detection setting with k machines, where machines receive samples $X = \{x_1, \ldots, x_k\}$ from either D_0 or D_1. Then for any communication protocol with transcript $\Pi(X)$ that solves the detection problem,

$$I\left(X; \Pi(X)\right) = \Omega(1), \text{ when } X \text{ is sampled from } D_0.$$

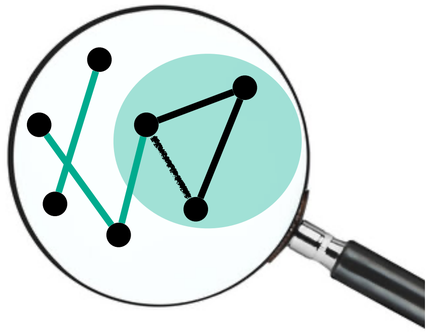
Non-trivial:

- If $t = 1$, $\mu_0 = \mathrm{Ber}(0.5)$, $\mathbb{E}_{\theta\sim P}[\mu_\theta]$ = always 1, then $I(X, \mathrm{AND}(X)) = 2^{-k}$.
- Yet, we show any successful protocol must have ***constant*** mutual information.

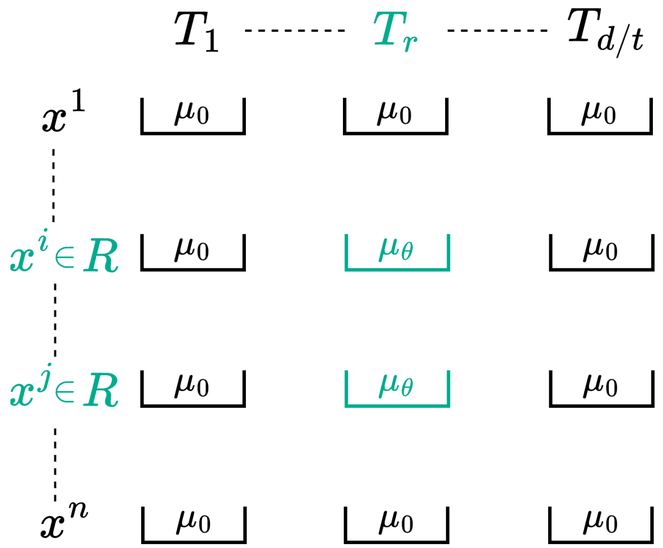
- We then apply **direct sum**, and lift IC bounds via **multi-pass IC notion** of [Braverman et al '24]

Conclusion

Conclusion



Detecting **planted structure** is common to many classic inference problems.



We provide a unified approach to proving **memory-sample tradeoffs** for planted structure detection

planted clique, Gaussian mean, sparse PCA, …

See our paper for more.

Better guarantees beyond $\log(d)$ *memory regime?*

Distinguishing vs Detection?