Pretrained transformer efficiently learns low-dimensional target functions in-context

Kazusato Oko, Yujin Song, Taiji Suzuki, Denny Wu

NeurIPS 2024 @ Vancouver Convention Centre

UC Berkeley UTokyo AIP NEW YORK UNIVERSITY FLATIRON INSTITUTE

In-Context Learning (ICL)

- Pretrained transformers can recognize patterns from prompts *without updating model parameters*

- A very short context can be sufficient

Please guess the number that fits in the '?'.

context
1,1 -> 2
2,3 -> 5
8,13 -> 21
6,0 -> 6
10,1 -> 11

query
5,27 -> ?

Question

The pattern in the given pairs of numbers appears to be the sum of the two numbers.

So, the number that fits in the '?' is 32.

ChatGPT (GPT-4)

Prior Works

- Known fact: linear transformers can emulate *linear regression on the context* in its forward pass [ACDS23, MHM23, ZFB23…]
  - requires the same context length (N) as the amount of data needed for linear regression
  - higher vector dimension (of $\boldsymbol{x}$)
    → higher required context length

prompt t

context:

$$\boldsymbol{x}_1 \to y_1 = \boldsymbol{x}_1^\top \boldsymbol{\beta}^t$$

$$\boldsymbol{x}_2 \to y_2 = \boldsymbol{x}_2^\top \boldsymbol{\beta}^t$$

$$\vdots$$

$$\boldsymbol{x}_N \to y_N = \boldsymbol{x}_N^\top \boldsymbol{\beta}^t$$

query:

$$\boldsymbol{x}^q \to y^q = ?$$

Q: Can TF outperform learning algorithms applied directly to the prompt?

(in terms of context length)

Our Main Message

Q: Can TF outperform learning algorithms working directly on prompt?

A: Yes, by *adapting to the problem structure* during pretraining

Problem Setting

- Learning single-index functions in-context
- On t-th prompt, $\boldsymbol{x} \sim N(0, \boldsymbol{I}_d) \in \mathbb{R}^d$ and

$$y = \sigma_*^t(\boldsymbol{x}^\top \boldsymbol{\beta}^t)$$

y depends only on the direction of $\boldsymbol{\beta}^t$

  - σ_*^t : random polynomial of degree P (nonlinear)
  - $\boldsymbol{\beta}^t \in \mathbb{R}^d$: random vector drawn from $r \ll d$-dimensional subspace of $\mathbb{R}^d$

Problem distribution is low-dimensional

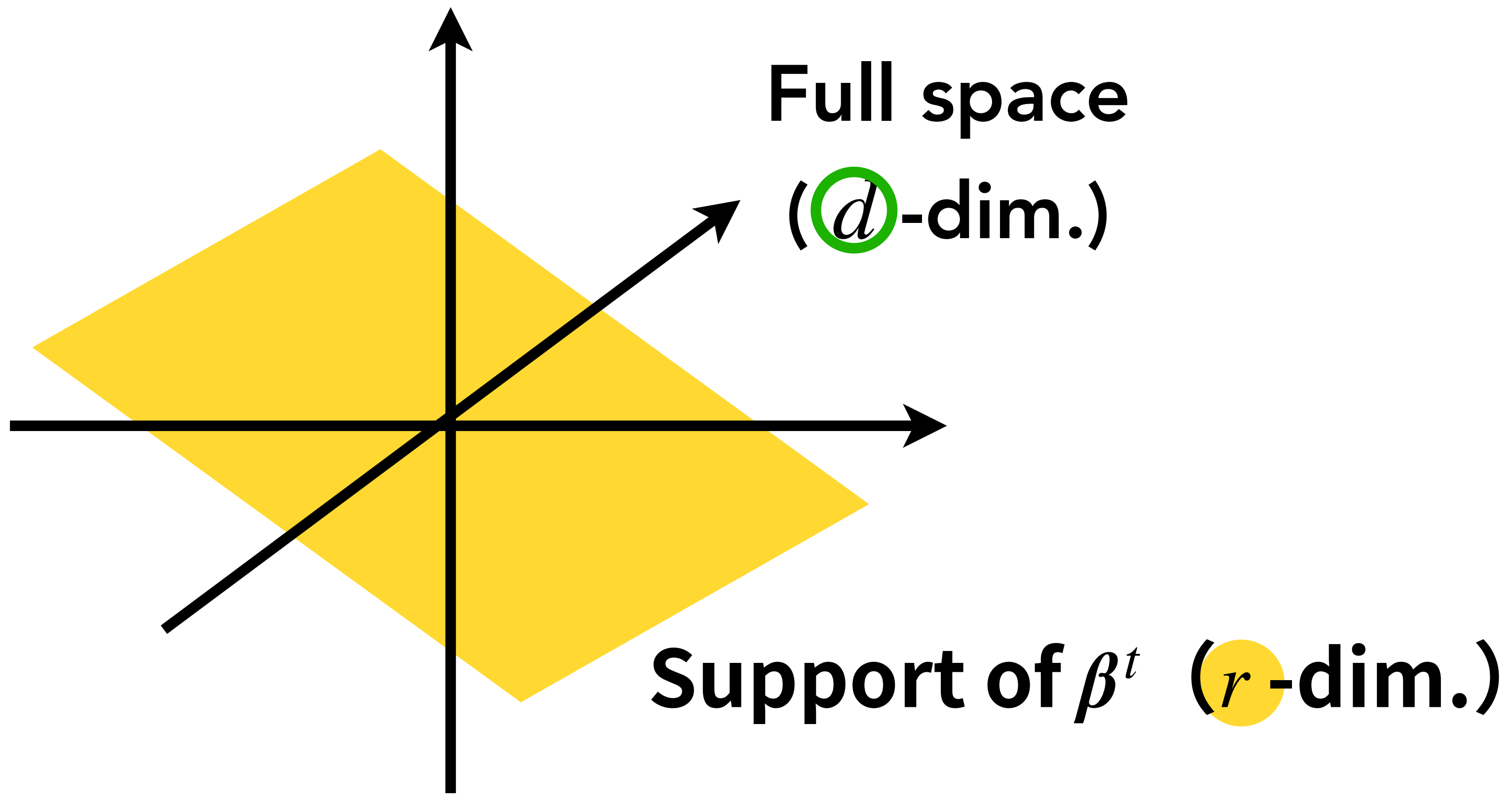
prompt t

context

$$\boldsymbol{x}_1 \to y_1 = \sigma_*^t(\boldsymbol{x}_1^\top \boldsymbol{\beta}^t)$$
$$\boldsymbol{x}_2 \to y_2 = \sigma_*^t(\boldsymbol{x}_2^\top \boldsymbol{\beta}^t)$$
$$\vdots$$
$$\boldsymbol{x}_N \to y_N = \sigma_*^t(\boldsymbol{x}_N^\top \boldsymbol{\beta}^t)$$

query

$$\boldsymbol{x}^q \to y^q = ?$$



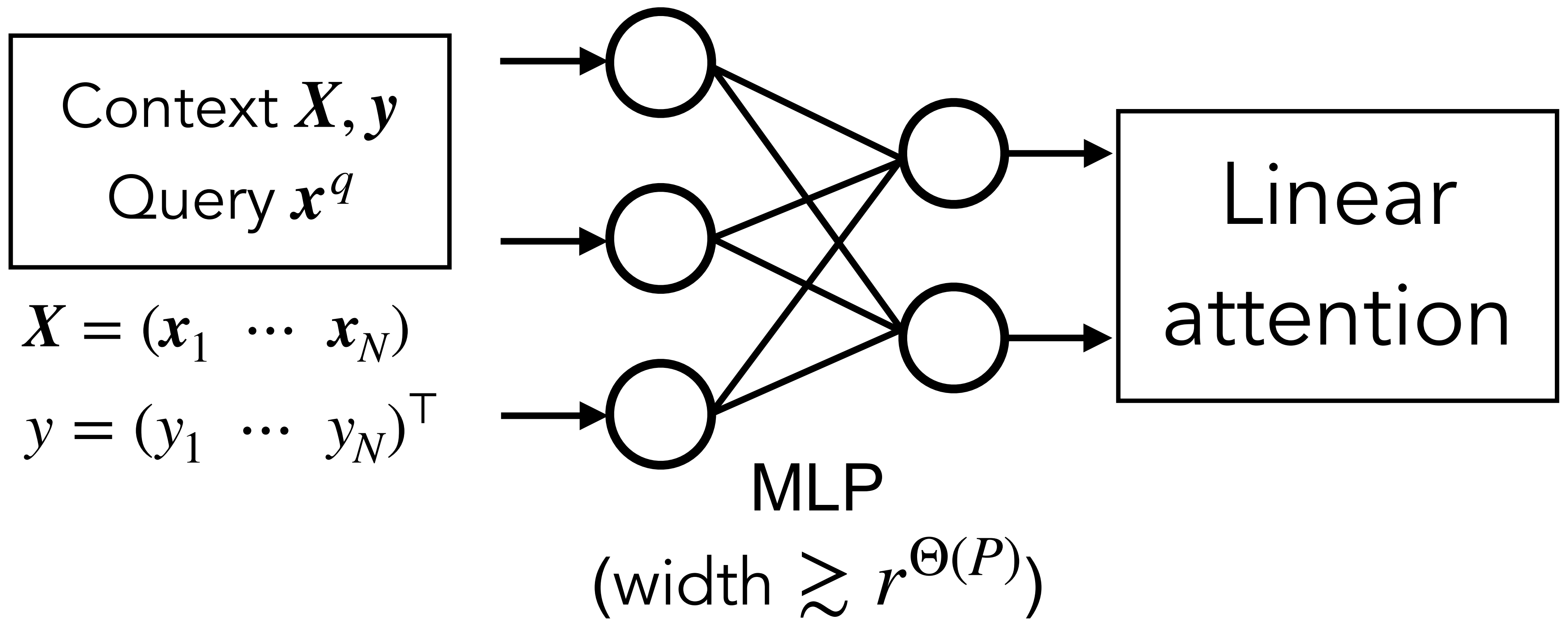
  - Learning algorithms (kernel, NN…) on the test prompt need poly(d) samples

…Can pretrained TF outperform them?

[GMMM21, BAGJ21]

Our Main Result

- Consider pretraining a single-layer transformer (nonlinear MLP+attention) on $d^{\Theta(Q)}$ tasks with a prompt length of $d^{\Theta(Q)}$ (Q: *lowest* degree of σ_* in $y = \sigma_*(\boldsymbol{x}^\top \boldsymbol{\beta})$).
Polynomial



$\hat{y}^q(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{x}^q)$

Estimation of y^q

Pretraining

I. One-step gradient descent on MLP weight

II. Ridge regression on attention matrix

- **Theorem** TF pretrained above achieves low test error ($\mathbb{E}[|\hat{y}^q - y^q|] = o_d(1)$) if context length N^* at test prompt satisfies $N^* \gtrsim r^{4P}$ (P: *highest* degree of σ_*)

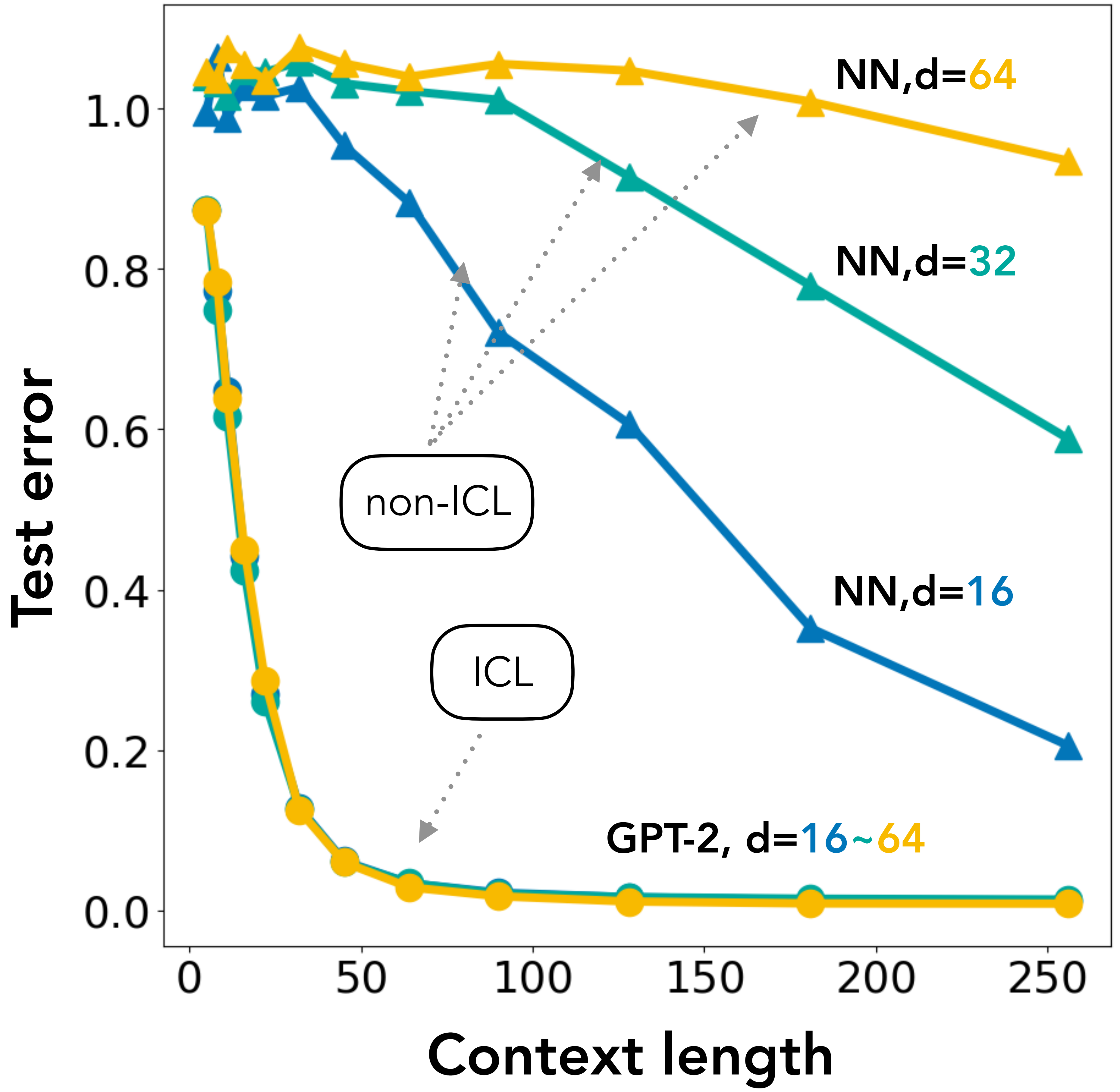
Required prompt length only depends on the inner dimension r

Baseline algorithms (kernel, NN) require d-dependent amount of data → superiority under $r \ll d$

*Pretraining is nonconvex optimization… end-to-end optimization & generalization analysis

Experiment

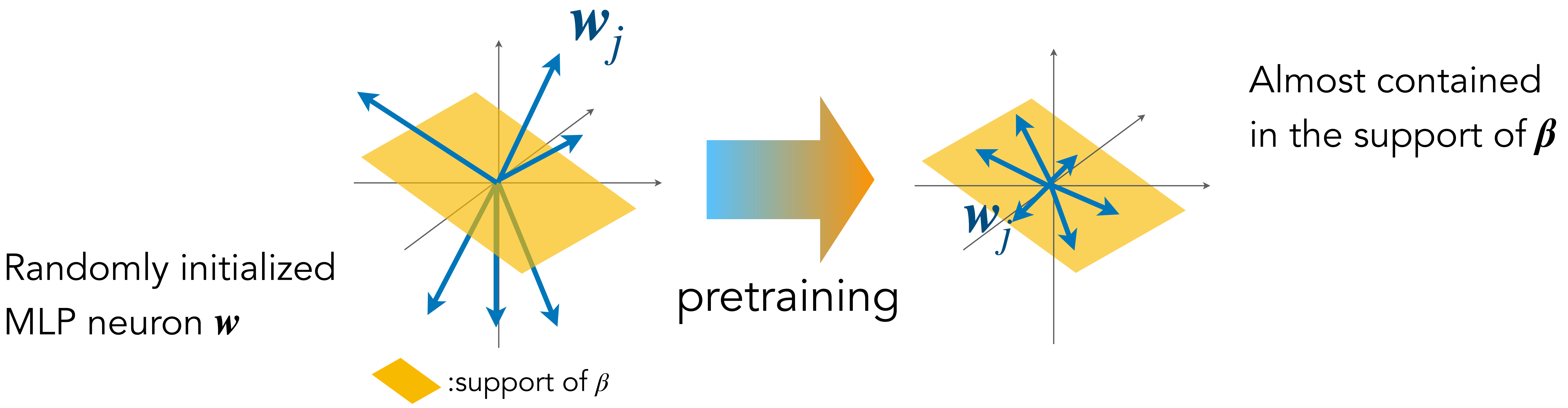
- We fix the inner dimension $r = 8$, while altering the ambient dimensiom d from 16 to 64, for the problem $y = \sigma_*^t(\boldsymbol{x}^\top \boldsymbol{\beta}^t)$.
- NN performance deteriorates with increasing d
- GPT-2 achieves low test error even when d is high



Takeaway & Mechanism

- Takeaway: TF can adapt to the prior distribution of problems via pretraining

- Mechanism: pretrained MLP neurons align with the r-dimensional subspace



- This "memorization" of the prior distribution of problems results in d-free context length complexity

See you in Vancouver!

preprint: https://arxiv.org/abs/2411.02544