上海交通大学
约翰·霍普克罗夫特
计算机科学中心
John Hopcroft Center for Computer Science

上海创智学院
SHANGHAI INNOVATION INSTITUTE
2024

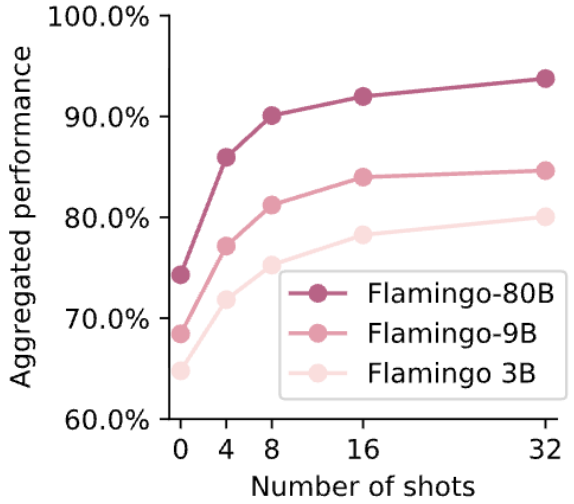
Towards a Reliable Theory of In-context Learning

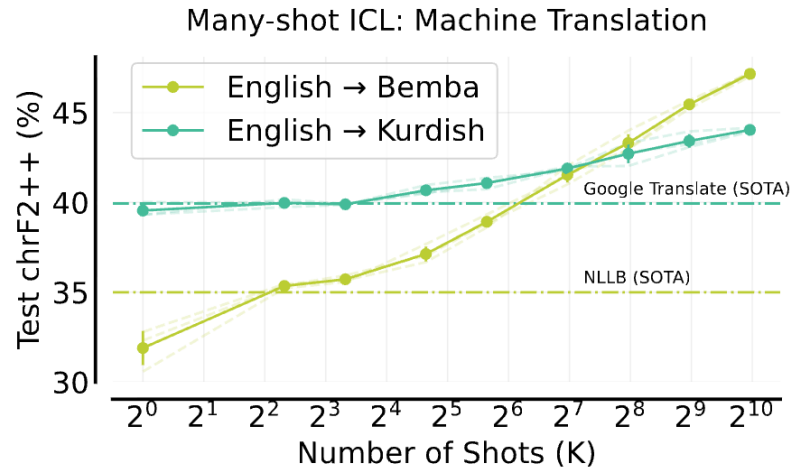
Shuai Li

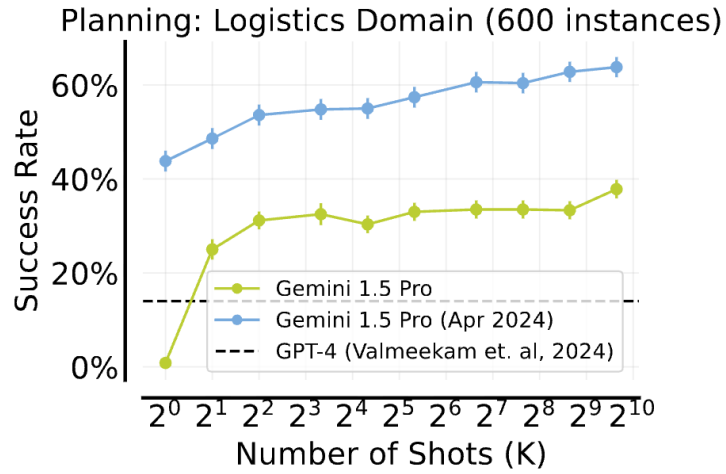
2026.08.04

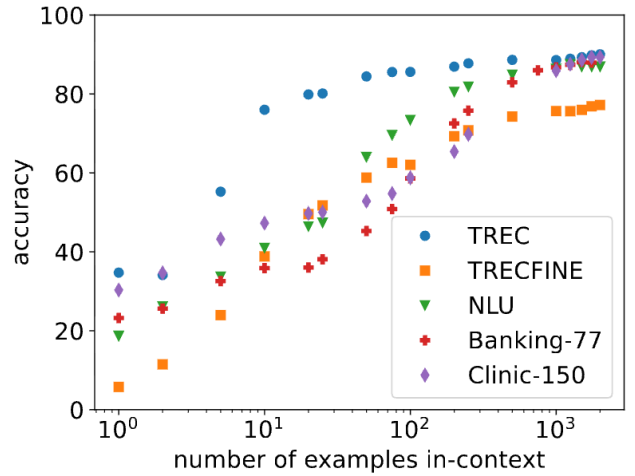
Seeing More Shots Improve Performance

- Multimodal: Flamingo [1]
- Machine Translation [2]
- Planning [2]
- Long-context classification [3]





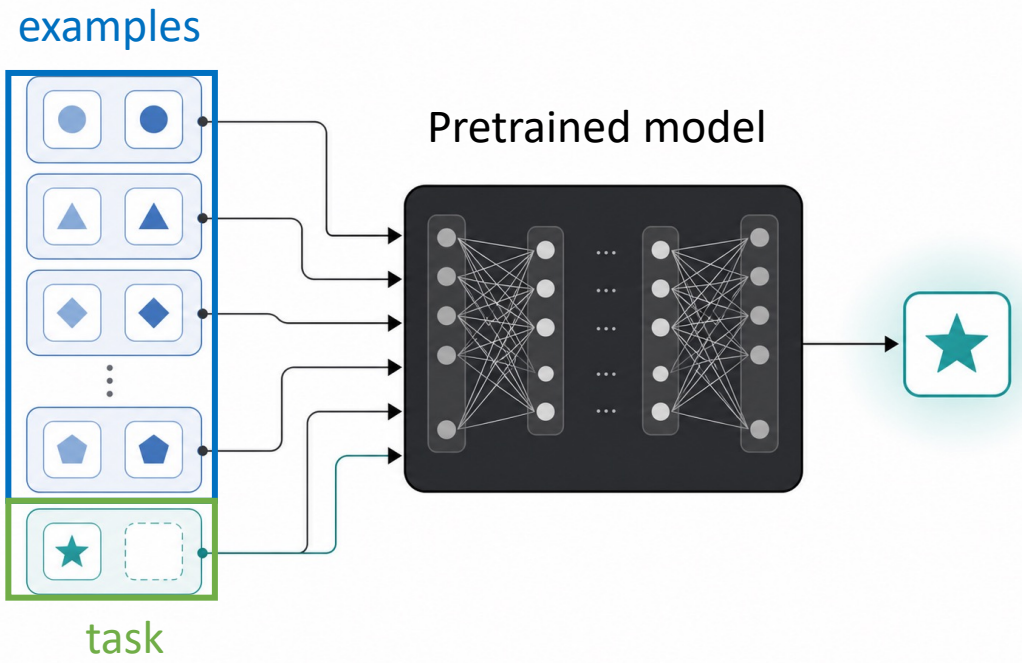




[1] ADLM+, Flamingo: a Visual Language Model for Few-Shot Learning, NeurIPS 2022.
[2] ASZB+, Many-Shot In-Context Learning, NeurIPS 2024.
[3] BIXA+, In-Context Learning with Long-Context Models: An In-Depth Exploration, ACL 2025.

In-context Learning (ICL)

- In-context learning: The model learns the task from the context of examples
- The output adapts to examples
- But the parameters of the model do not change
- How does the learning happen?



Models learn from examples during inference

- BMRS+, Language Models are Few-Shot Learners, NeurIPS 2020.

Outline

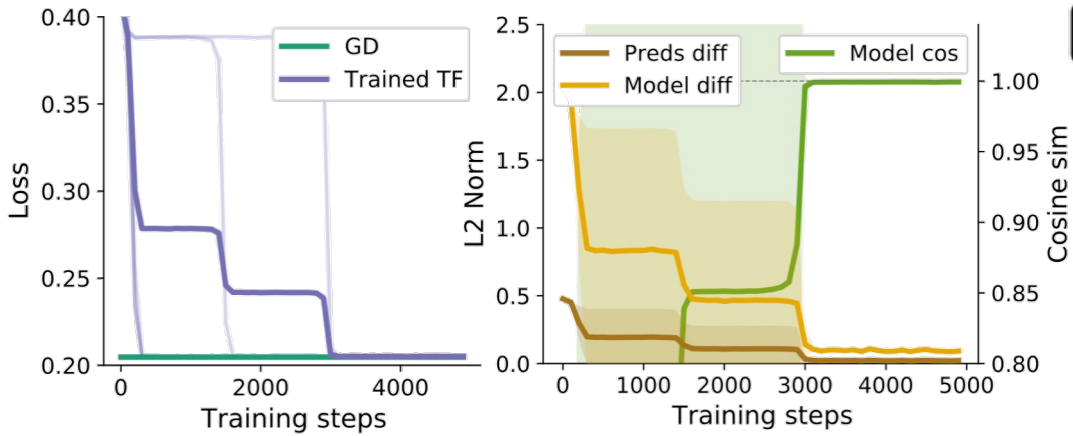
- Transformer learns gradient implicitly
- ICL extends to state-space models
- The effect of position encoding on ICL

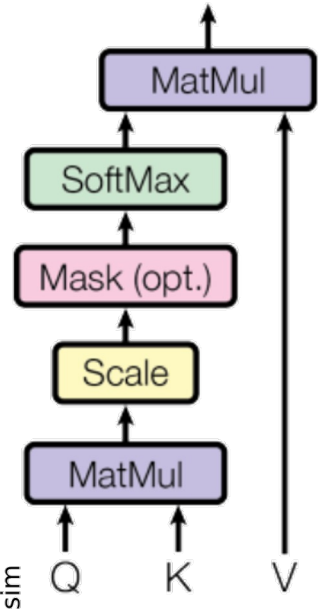
Transformer Learns Gradient Implicity

- Previous work studies linear self-attention for linear regression task
- $\mathrm{LSA}_\theta(X) = XW_QW_K^\top X^\top XW_V$
- Linear Regression $L(W) = \sum_i \|Wx_i - y_i\|^2$
- Theorem. $\exists\theta$ s.t. $\mathrm{LSA}_\theta(X) = Y_{\mathrm{GD}}$ where

$$\sum_i \|(W - \nabla W)x_i - y_i\|^2 = \sum_i \|Wx_i - (y_i + \Delta y_i)\|^2$$

- Trained LSA ≈ Constructed LSA





Transformer Attention

- ONRSMZV, Transformers Learn In-Context by Gradient Descent, ICML 2023.

More Realistic Softmax Attention

- No closed-form construction anymore!
- Softmax Attention $\mathrm{SA}_\theta(X) = Z^{-1}\exp(XW_QW_K^\top X^\top)\,XW_V$
- Softmax regression $L(W;X,Y) = \|\langle \exp(XW)\,, \mathbb{I}_n\rangle^{-1}\exp(XW) - Y\|$
- Theorem. Weight shift ≈ Data shift w/ same Lipschitz continuity

$$L(W;X,Y)$$

$$L(W+\Delta W;X,Y) \approx L(W;X,Y+\Delta Y) \approx L(W;X+\Delta X,Y)$$

Theorem. $\exists\theta$ s.t. $\mathrm{LSA}_\theta(X) = Y_{\mathrm{GD}}$ where

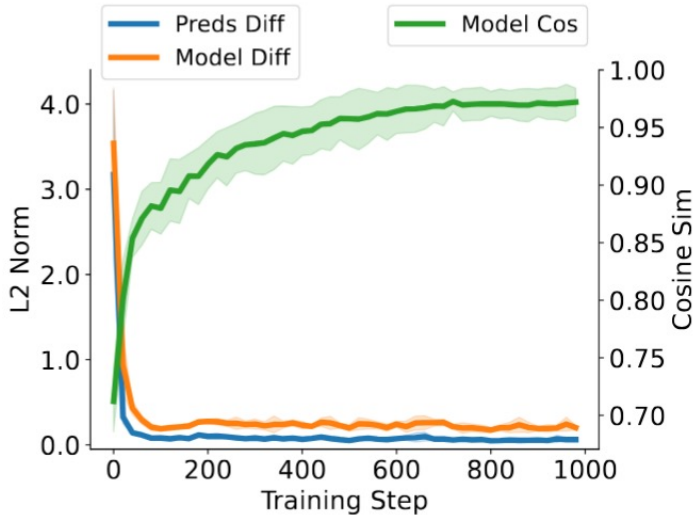
$$\sum_i \|(W+\Delta W)x_i - y_i\|^2 = \sum_i \|Wx_i - (y_i + \Delta y_i)\|^2$$

- LSXYZ, The Closeness of In-Context Learning and Weight Shifting for Softmax Regression, NeurIPS 2024.

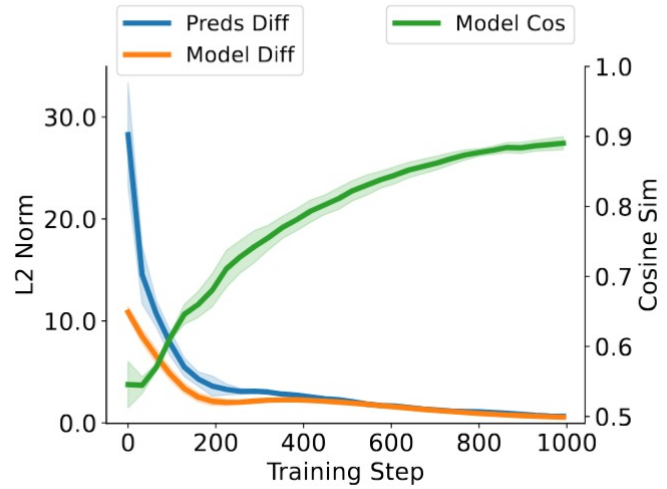
More Realistic Softmax Attention:Experiments

- Trained SA ≈ one-step GD for softmax regression

- Preds Diff: $\|\hat{y}(X) - \tilde{y}(a)\|_2$
- Model Diff: $\left\|\frac{\partial \hat{y}(X)}{\partial X} - \frac{\partial \tilde{y}(a)}{\partial a}\right\|_2$
- Model Cos: $\mathrm{CosSim}\left(\frac{\partial \hat{y}(X)}{\partial X}, \frac{\partial \tilde{y}(a)}{\partial a}\right)$



context length
n=200

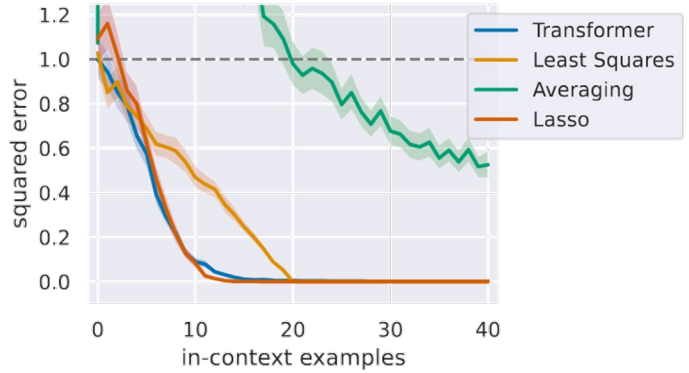


n=1000

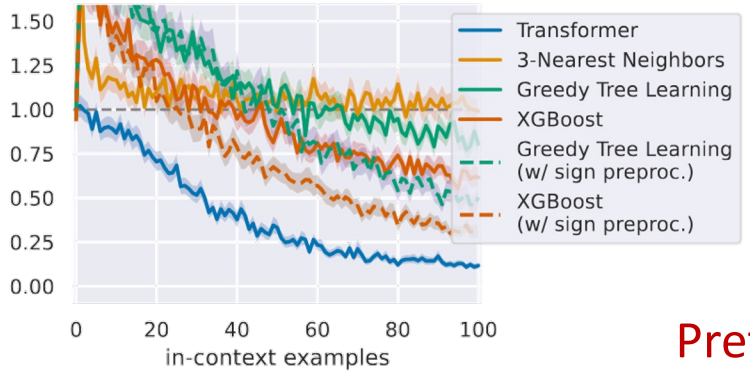
- LSXYZ, The Closeness of In-Context Learning and Weight Shifting for Softmax Regression, NeurIPS 2024.

How about General Tasks?

- ICL is observed for many tasks

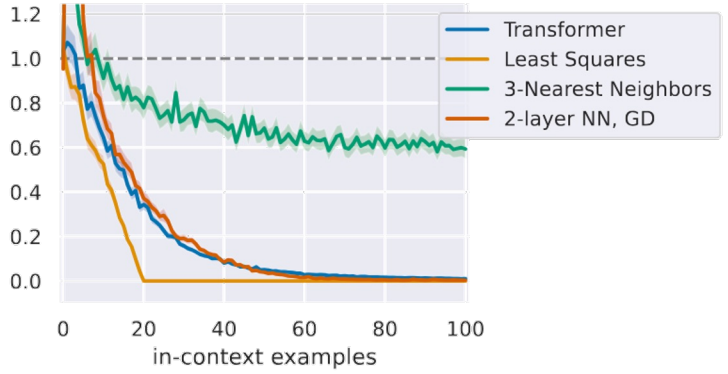


(a) Sparse linear functions



(b) Decision trees

(c) 2-layer NN



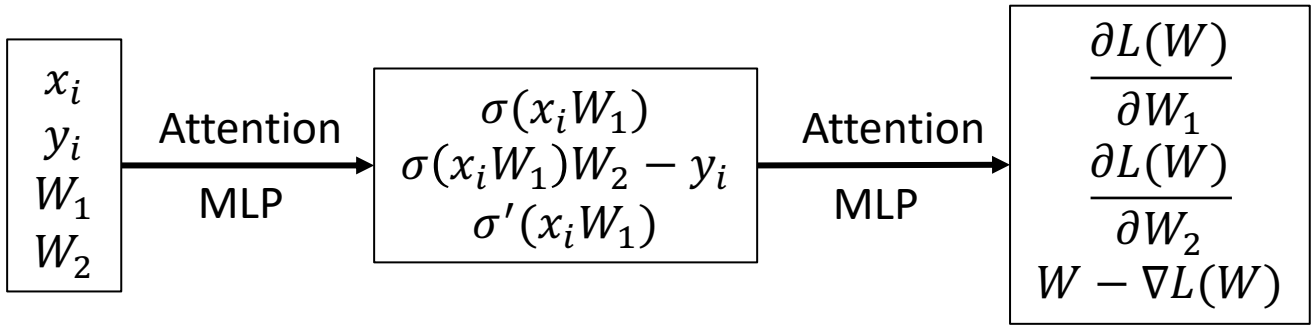
(d) 2-layer NN, eval on linear functions

Pretrained 12-layer, 8-head Transformers can learn various functions in-context

- GTLV, What Can Transformers Learn In-Context? A Case Study of Simple Function Classes. NeurIPS 2022.

Constructed TF can IC-Learn 2-layer NN

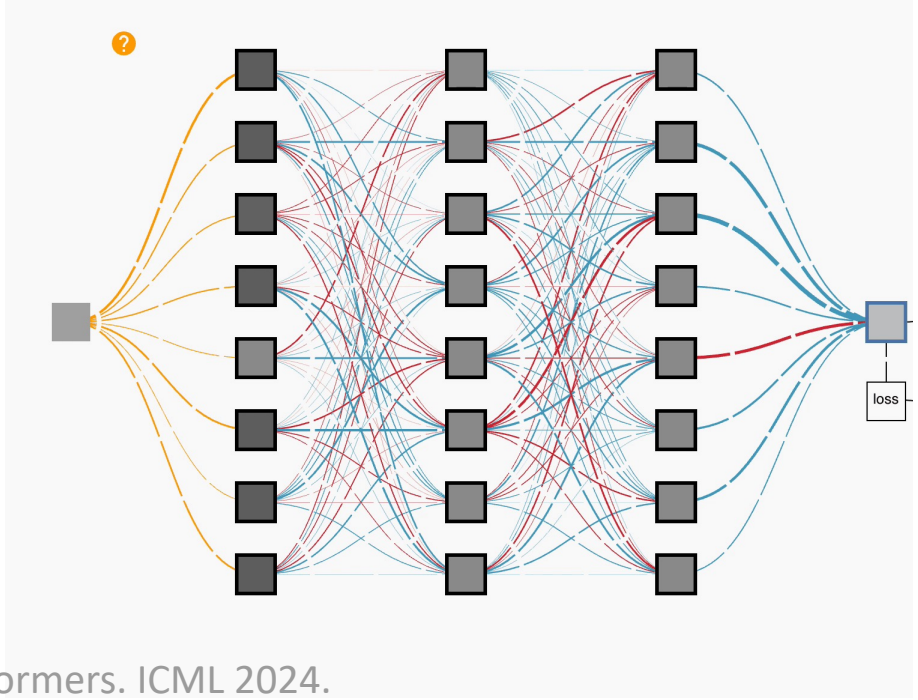
- Existing work considers 2-layer NN regression task
- Regression $L(W) = \|\sigma(XW_1)W_2 - Y\|^2$
- $\frac{\partial L(W)}{\partial W_1} = 2\sum_i(\sigma(x_iW_1)W_2 - y_i) \cdot \sigma'(x_iW_1)W_2 \cdot \mathbf{1} \cdot x_i$
- $\frac{\partial L(W)}{\partial W_2} = 2\sum_i(\sigma(x_iW_1)W_2 - y_i) \cdot \sigma(x_iW_1)$
- Theorem. Constructed transformer ≈ GD of 2-layer NN task



- BCWXM, Transformers as Statisticians: Provable In-Context Learning with In-Context Algorithm Selection. NeurIPS 2023.

General Tasks

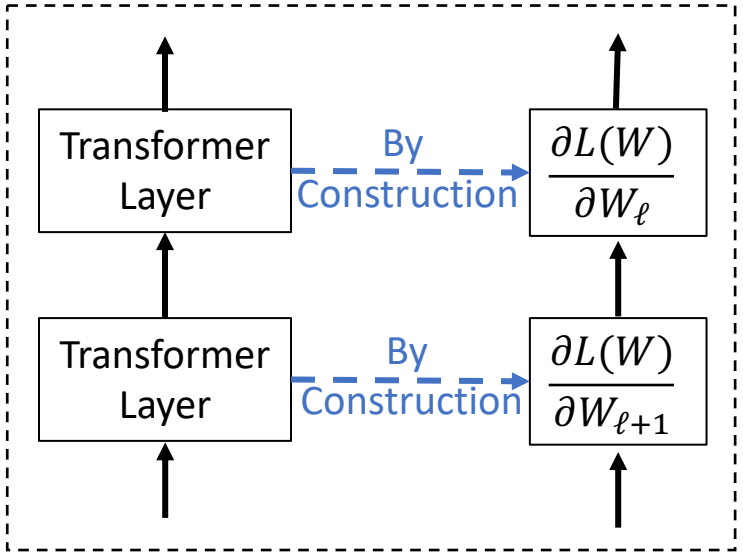
- Previous explicit construction fails for general tasks
- General regression $L(W) = L(\sigma(\sigma(\sigma(XW_1)\cdots)W_{n-1})W_n, Y)$
- $\nabla L(W)$ has higher-order terms $\sigma'(\sigma'(\sigma'(XW_1)\cdots)W_{n-1})W_n$ cannot be represented by quadratic $QK^\top V$
- Directly compute gradient will relate to exponentially many elements (parameters, hiddens)
- $\Longrightarrow$cannot explicitly construct
- Idea: Think backpropagation!



- WJL, In-context Learning on Function Classes Unveiled for Transformers. ICML 2024.

Layered Transformer Approximate General GD

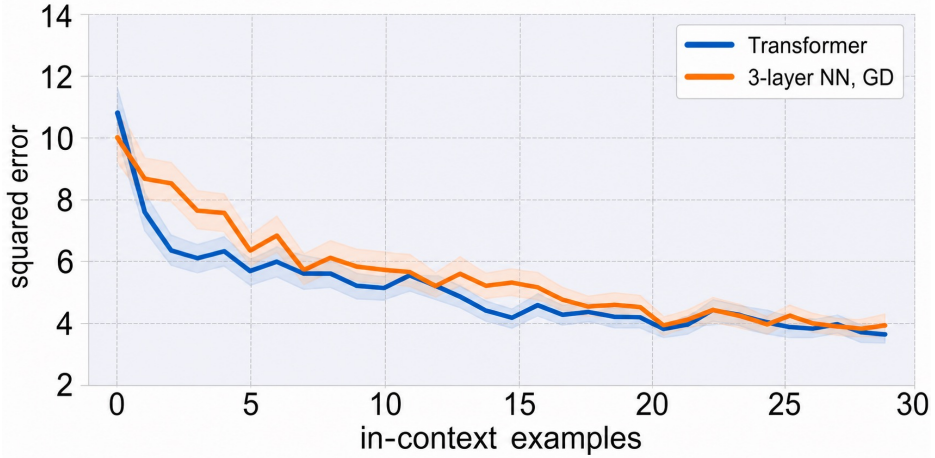
- Compute in a recursive way
- one layer Transformer approximate one-layer GD
- Then stack them up



- Theorem. For n-layer, width-K NN regression task, $\exists\ O(n)$-layer Transformer that can approximate a one-step GD on the regression task

$$W^+ = W - \eta\big(\nabla L(W) + \epsilon(W)\big)$$

where $H \le O(K^2\epsilon^{-2}), d \le O(nK^2)$



first extend to 3-layer NN regression task

- WJL, In-context Learning on Function Classes Unveiled for Transformers. ICML 2024.

General Task Example: Indicator Function

- Regression target $\mathbb{I}(\|Ax+b\| \leq r)$
- If approximated 2-layer NN, requires $O(\exp(\delta^{-0.25}))$-width NN
- If approximated 3-layer NN, requires $O(\delta^{-0.5})$-width

- Theorem. The constructed $O(\log(1/\epsilon))$ TF layers can learn the indicator function with accuracy $O(\epsilon+\delta)$, and $\max\{H,d\} \leq O(\delta^{-1}\epsilon^{-2})$.

- WJL, In-context Learning on Function Classes Unveiled for Transformers. ICML 2024.

Construction Algorithm Adapt to Smoothness

- Regression target $\in \mathcal{C}^s([0,1]^D)$
- Lemma. An $O(\log(1/\delta))$-layer, $O\left(\delta^{-D/s}\right)$-width NN can approximate it w/ accuracy δ.

- Theorem. The constructed $O(\log(1/\epsilon)\log(1/\delta))$ TF layers can learn the s-smooth function w/ accuracy $O(\epsilon+\delta)$, and $\max\{H,d\} \leq \tilde{O}\left(\delta^{-2D/s}\epsilon^{-2}\right)$.

- WJL, In-context Learning on Function Classes Unveiled for Transformers. ICML 2024.

Overview on the Results

ICL task	Attention	Algorithm	layer	heads
Linear Regression [1]	linear	GD	1 / GD	1
Linear Regression [2]	softmax	GD / closed form solution	9 / GD	2
2-layer NN + Generalized Linear Regression [3]	ReLU	approximate GD	2 / GD	$O(K\epsilon^{-2})$
Ridge regression with feature representation [4]	ReLU	GD	1 / GD	5
n-layer NN + indicator/smooth functions [5]	general activation	approximate GD	n / GD	$O(K^2\epsilon^{-2})$
n-layer NN [6]	softmax	approximate GD through CoT	1-layer with N CoT steps	/

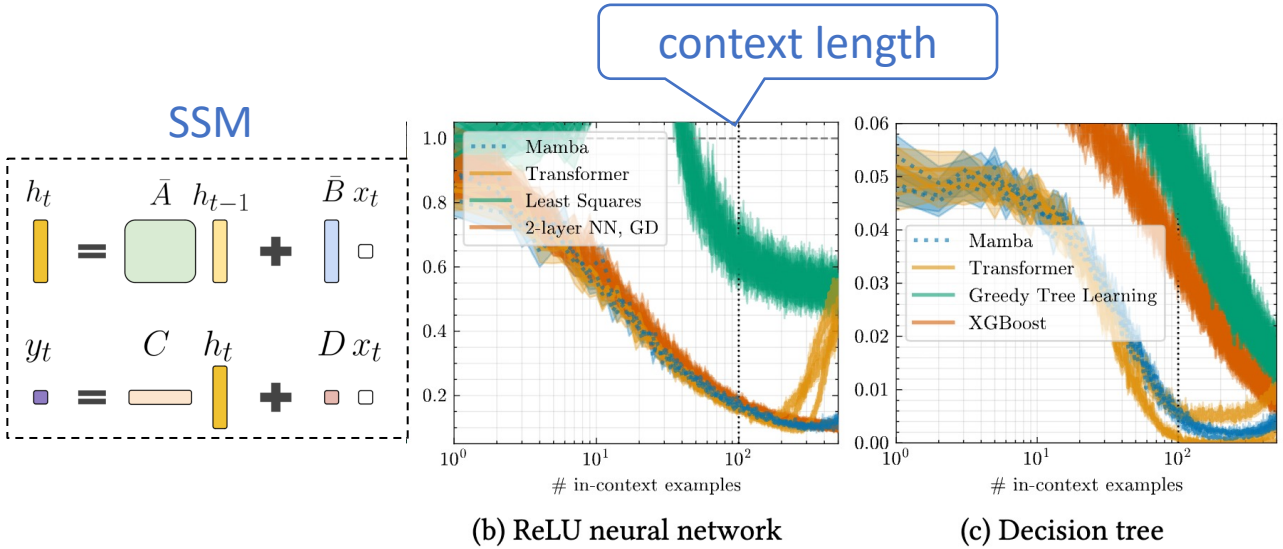
[1] ONRSMZV, Transformers Learn In-context by Gradient Descent. ICML 2023.
[2] ASAMZ, What Learning Algorithm is In-context Learning? Investigations with Linear Models. ICLR 2023.
[3] BCWXM, Transformers as Statisticians: Provable In-Context Learning with In-Context Algorithm Selection. NeurIPS 2023.
[4] GHMWXSB, How Do Transformers Learn In-Context Beyond Simple Functions? A Case Study on Learning with Representations. ICLR 2024.
[5] WJL, In-context Learning on Function Classes Unveiled for Transformers. ICML 2024.
[6] CGLHL, Chain-of-Thought Gradient Descent. ICML 2026.

Outline

- Transformer learns gradient implicitly
- ICL extends to state-space models
- The effect of position encoding on ICL

ICL extends to State-space Models

- Inference of transformer requires $O(n^2)$ computation
- Consider state-space models, like Mamba [1]
- Mamba shows ICL properties [2], but worse than transformer [3]
- Mamba-transformer hybrid structure is better than each [3]



(b) ReLU neural network (c) Decision tree

Number of queries	4		16	
Embedding dimension	64	128	64	128
Mamba	8.64e-1	1.64e-1	7.23e-1	1.50e-1
Transformer without PE	1.14e-3	8.66e-5	7.61e-5	5.55e-5
Transformer with PE	**5.17e-6**	**8.76e-7**	3.99e-5	**2.46e-7**

	Transformer	Mamba	MambaFormer
Linear regression	✓	✓	✓
Sparse linear regression	✓	✓	✓
2NN regression	✓	✓	✓
Decision Tree	✓	▲	✓
Orthogonal-outlier regression	✓	▲	✓
Many-outlier regression	▲	✓	✓
Sparse parity	✗	✓	✓
Chain-of-Thought I/O	✓	✓	✓
Vector-valued MQAR	✓	✗	✓

[1] GD, Mamba: Linear-Time Sequence Modeling with Selective State Spaces. arXiv:2312.00752.
[2] GSSGH, Is Mamba Capable of In-Context Learning? AutoML 2024.
[3] PPXL+, Can Mamba Learn How to Learn? A Comparative Study on In-Context Learning Tasks. ICML 2024.

Comparison of Mamba & Transformer

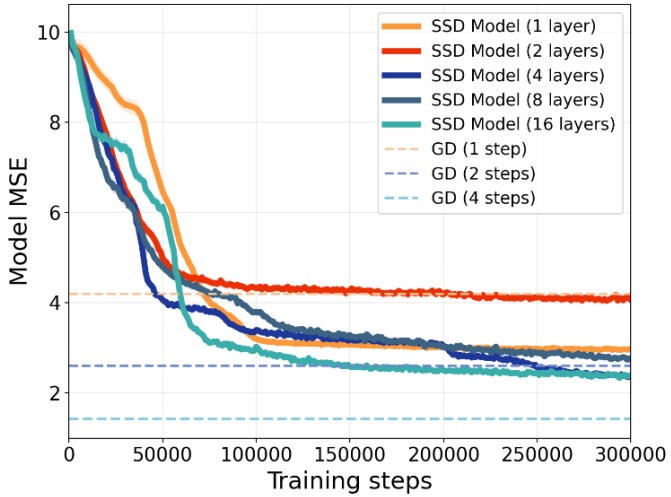
- Mamba adopt diagnal A and structured mask:
  State Space Duality [1] $\mathrm{SSD}_\theta(X) = [L \circ (XW_BW_C^\top X^\top)]X$

learnable mask

- Note $\mathrm{LSA}_\theta(X) = \left[M \circ XW_QW_K^\top X^\top\right] XW_V$

fixed mask

value



16 layers of SSD can't even match 4 step GD!

- For linear regression task $L(W)$,
  recall n-layer LSA can perform n-step GD of the task
- Proposition. n SSD layers can't perform n-step GD of linear regression task

[1] DG, Transformers are SSMs: Generalized Models and Efficient Algorithms Through Structured State Space Duality. ICML 2024.
[2] WL, In Context Stochastic Gradient Descent with Hybrid Mamba-2. In submission.

Hybrid Layer Can Achieve SGD

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \quad L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

causal mask can only select full batch

learnable mask can select mini batch

- LSA can perform full-batch GD since its mask can only select all samples could not select mini-batch
- SSD has learnable mask that can select mini-batch, but will modify inputs due to state compression. Can't perform gradient while keeping original input. Need independent value matrix to perform correct SGD update

- Theorem. For linear regression task and mini-batch $\mathcal{B}$, ∃ one hybrid layer LSA ∘ SSD perform one step of mini-batch SGD

$$W^+ = W - \eta \nabla L(W; \mathcal{B})$$

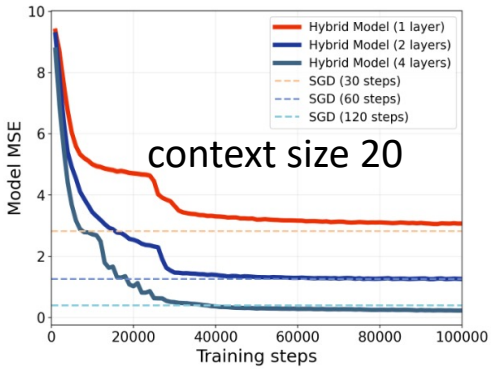
achieved by constructing learnable mask L of SSD.

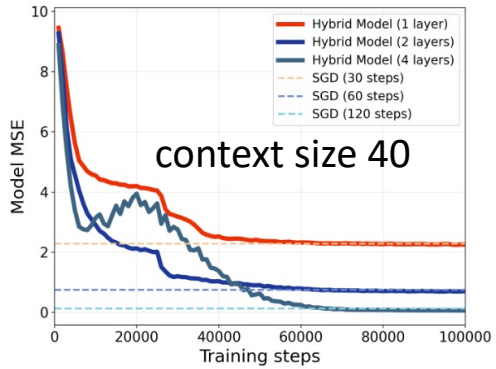
- WL, In Context Stochastic Gradient Descent with Hybrid Mamba-2. In submission.

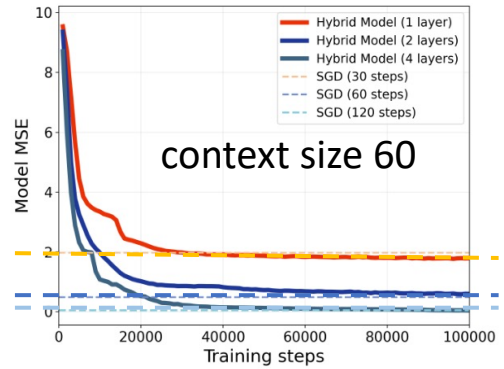
Experiments

- Compare 1-hybrid-layer (batch-1) vs. 30-step-SGD
  (2-hybrid-layer vs 60-step-SGD, 4-hybrid-layer vs 120-step-SGD)
- hybrid-layer converge to SGD-performance after training

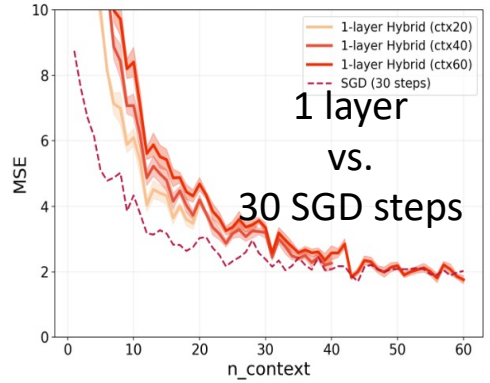
30 SGD steps
60 SGD steps
120 SGD steps

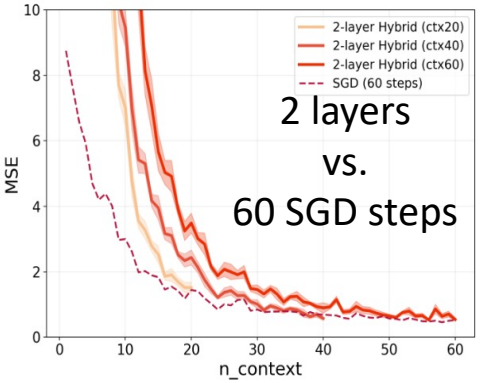


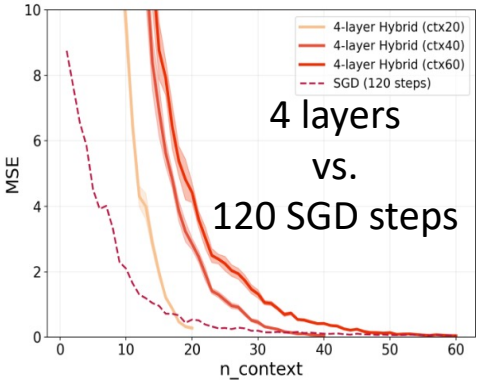




- hybrid-layer match to SGD-performance across context-length







- WL, In Context Stochastic Gradient Descent with Hybrid Mamba-2. In submission.

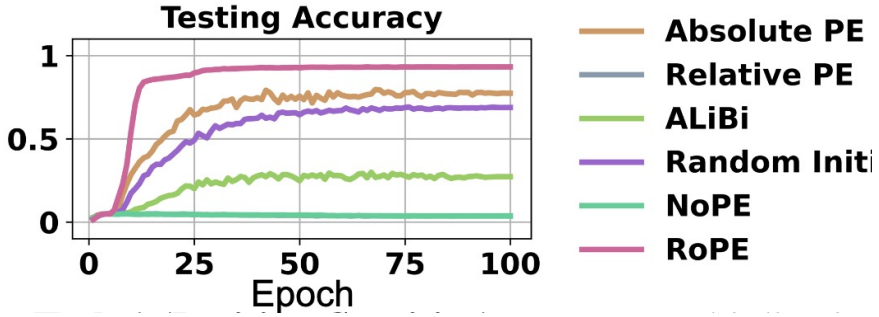
Outline

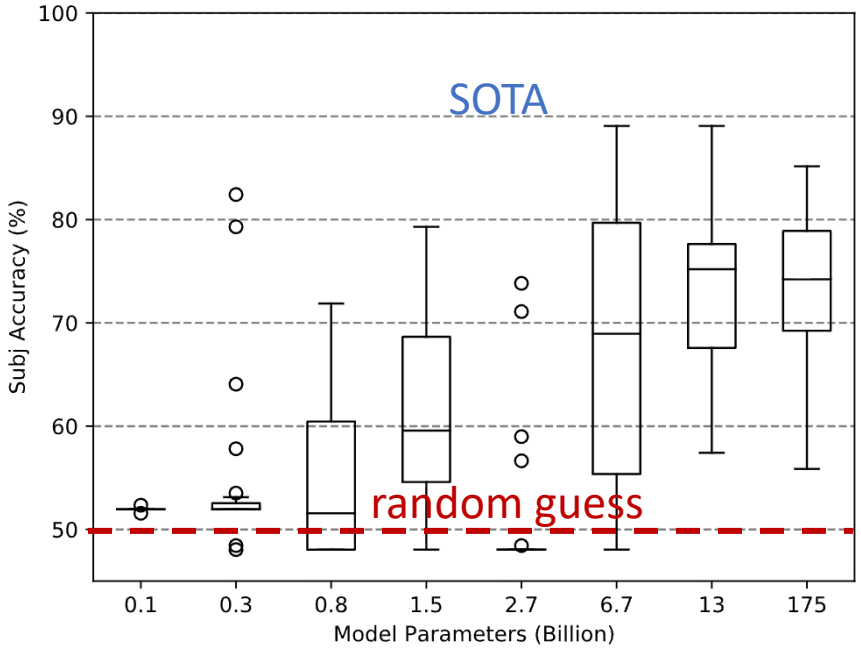
- Transformer learns gradient implicitly
- ICL extends to state-space models
- The effect of position encoding on ICL

Position Effect: Existing Work Fail to Explain

- ICL performance for different ordering of input vary from SOTA to random
- Studied attention $\mathrm{Attn}_\theta(X) = \sigma(XW_QW_K^\top X^\top)\,XW_V$ is permutation invariant for output
  $e_{N+1}^\top \mathrm{Attn}_\theta(PX) = \sigma\left(PXW_QW_K^\top X^\top P^\top\right) PXW_V = P\,\sigma\left(XW_QW_K^\top X^\top\right) P^\top PXW_V = e_{N+1}^\top P\mathrm{Attn}_\theta(X)$
  suppose P does not change query
- $X^\top = \begin{bmatrix} x_1 & x_2 & \cdots & x_N & x_{N+1} \\ y_1 & y_2 & \cdots & y_N & 0 \end{bmatrix}$

	MLM Perplexity
NoPos	147.18
Learned	4.06
Sinusoidal	4.07
ALiBi	4.00





- Then how to explain?

[1] HRPIL, ADLM+, Transformer Language Models without Positional Encodings Sti.. Learn Positional Information, NeurIPS 2022.
[2] GCZZH, Deconstructing Positional Information: From Attention Logits to Training Biases, ICLR 2026
[3] LBMRS, Fantastically Ordered Prompts and Where to Find Them. ACL 2022.

What is the Role of Position Encoding?

- Pratical work adopts position encoding to explicitly encode position information
- $e_{N+1}^{\top}\mathrm{Attn}_{\theta}(PX+E) \neq e_{N+1}^{\top}\mathrm{Attn}_{\theta}(X+E)$
- Position encoding makes ordering matter
- But how different!
- Consider two types of PE
  - one-hot concatenated PE
  - multiplicative RoPE for LLM

  and two types of tasks
  - Linear regression (order does not matter)
  - Dynamical system $X_{i+1} \approx AX_i$ (order matters)

RoPE

$$R_m = \begin{bmatrix} R(m\theta_0) & 0 & \cdots & 0 \\ 0 & R(m\theta_1) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R(m\theta_{d/2-1}) \end{bmatrix}$$

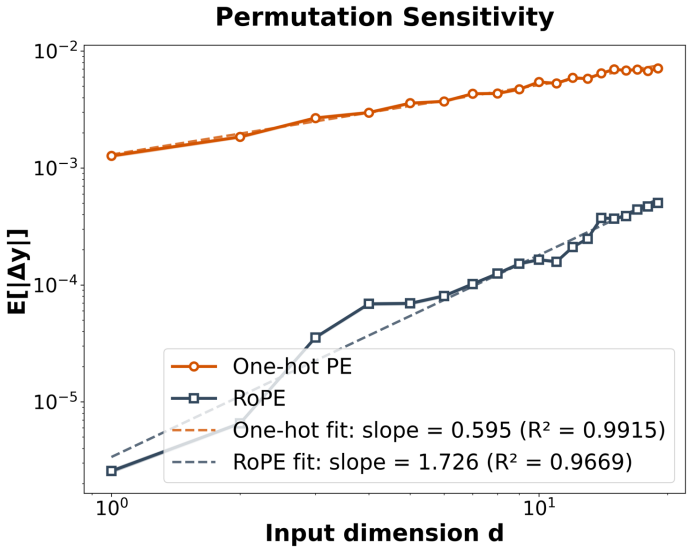
$$R(m\theta_r) = \begin{bmatrix} \cos(m\theta_r) & -\sin(m\theta_r) \\ \sin(m\theta_r) & \cos(m\theta_r) \end{bmatrix}$$

- SLPMWL, RoFormer: Enhanced Transformer with Rotary Positoin Embedding. Neurocomputing 2024.

Permutation Scaling

- Suppose permutation P can be decomposed into k transpositions
- $X^\top = \begin{bmatrix} x_1 & x_2 & \cdots & x_N & x_{N+1} \\ y_1 & y_2 & \cdots & y_N & 0 \end{bmatrix}$

k transpositions in total



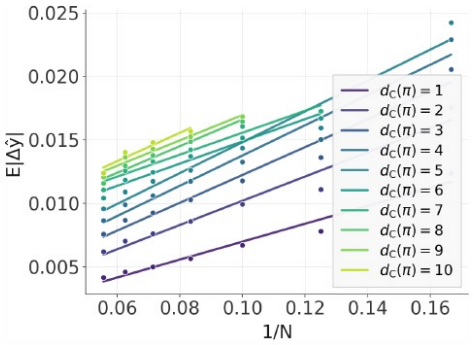
- Theorem. For linear regression task,
  - one-hot PE: $\mathbb{E}[|\mathrm{LSA}_\theta(PX+E) - \mathrm{LSA}_\theta(X+E)|] \le C_W d^{1/2} k/N$
  - Rope: $\mathbb{E}[|\mathrm{LSA}_\theta(PX+E) - \mathrm{LSA}_\theta(X+E)|] \le C_W d^{5/2} k/N$

  For dynamical system task,
  - one-hot PE: $\mathbb{E}[|\mathrm{LSA}_\theta(PX+E) - \mathrm{LSA}_\theta(X+E)|] \le C_A d^{3/2} k/N$
  - Rope: $\mathbb{E}[|\mathrm{LSA}_\theta(PX+E) - \mathrm{LSA}_\theta(X+E)|] \le C_A d^{7/2} k/N$
- Result for softmax regression is similar but different in role of $1/N$

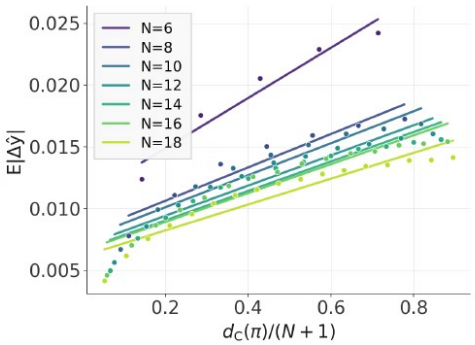
- WJL, Positional Encoding and Prompt-Order Sensitivity in In-Context Learning, in submission

Experiments

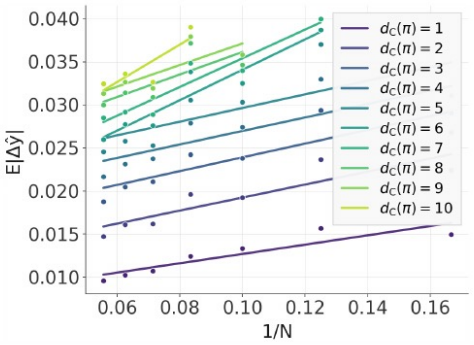
Linear regression



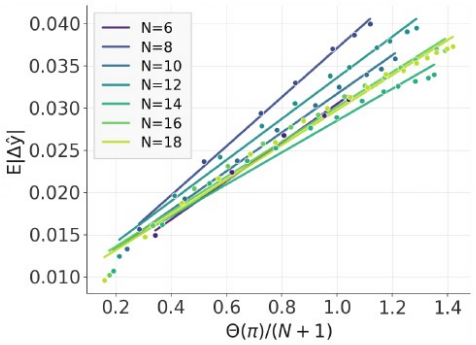
(a) One-hot, varying N



(b) One-hot, varying k



(c) RoPE, varying N

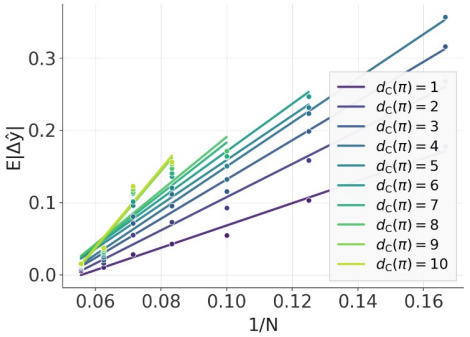


(d) RoPE, varying k

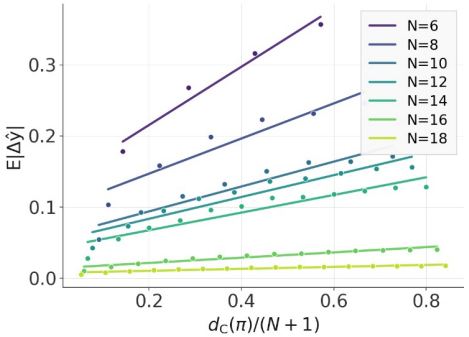
vary $1/N$

vary k

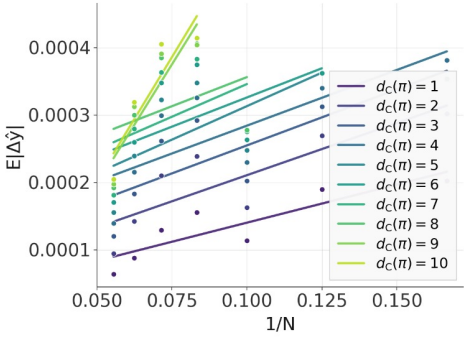
Dynamical system



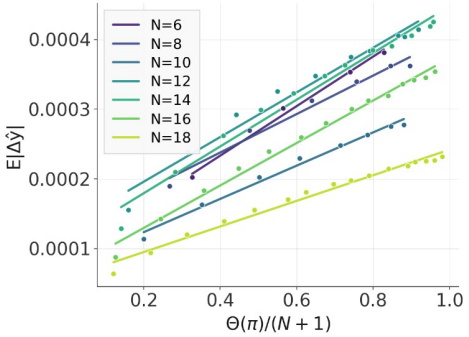
(a) One-hot, varying N



(b) One-hot, varying k



(c) RoPE, varying N



(d) RoPE, varying k

vary $1/N$ $O(k/N)$ law vary k

- WLJ, Positional Encoding and Prompt-Order Sensitivity in In-Context Learning, in submission

Conclusions

- Perspective: In-Context Adaptation Mirrors Weight Adaptation
  - Context shifts in softmax attention and weight shifts in softmax regression share the same Lipschitz stability
  - The predictions and learned models become closely aligned during training
- Algorithms: ICL Implements GD over Rich Function Classes
  - Transformers can learn DNNs in-context, thus approximating general functions
  - Mamba-Transformer hybrid models can implement SGD
- Robustness: Positional Encoding Drives Prompt-Order Sensitivity
  - Positional encoding breaks permutation invariance and changes ICL predicts
  - Similar sensitivity scaling emerges across tasks and encoding schemes

Future Work

- Perspective
  - Deep Transformers with MLP
- Algorithms
  - Identify the implicit function classes actually used by LLMs
  - Whether transformer models learn certain algorithms during pretraining phase
- Robustness
  - Extend the PE sensitivity theory to softmax attention and deep models with MLP
  - Design new PEs with robust prediction

Thanks!

Shuai Li

- Associate Professor
- Shanghai Jiao Tong University
- Research: RL/ML theory
- https://shuaili8.github.io/

Students:

Zhijie Wang

Yu Xia

Tianyi Zhou

Collaborators:

Bo Jiang

Zhao Song

Tong Yu

Questions?