Towards Trustworthy Foundation Models I:

Unveiling the Mystery behind In-Context Learning

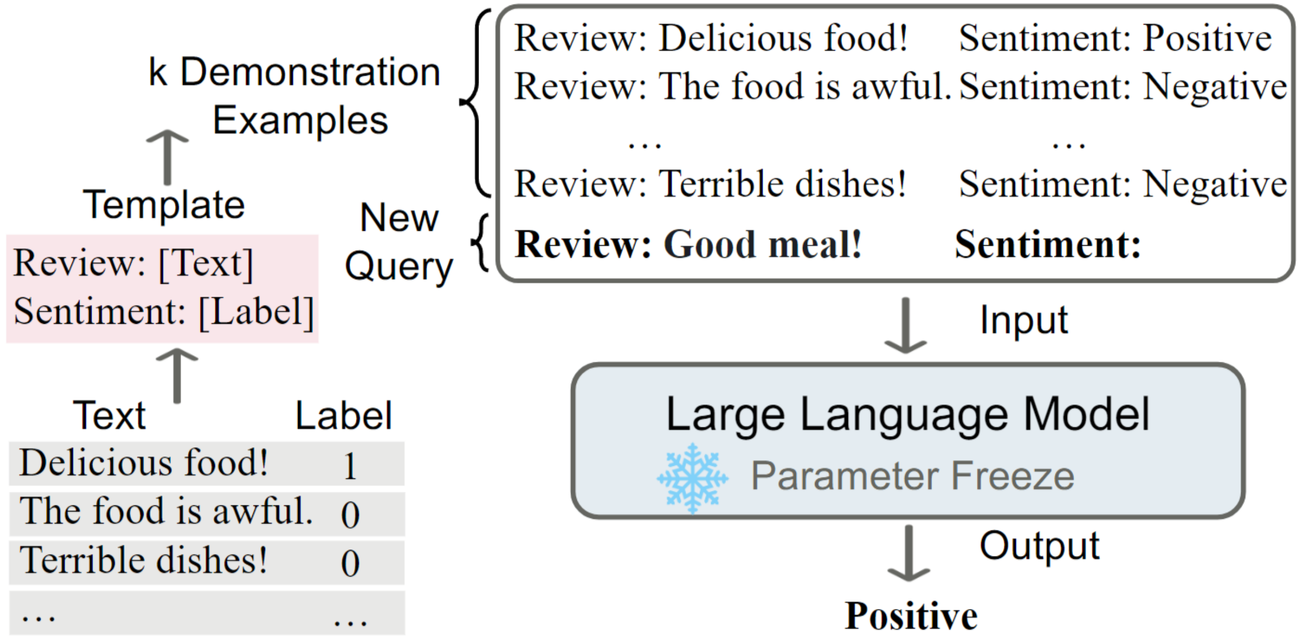
Chenyu Zheng

2024.5.30

Table of Contents

Definition of ICL

- ICL is the ability of foundation models to learn to perform downstream task based on the context without any explicit updates to parameters.

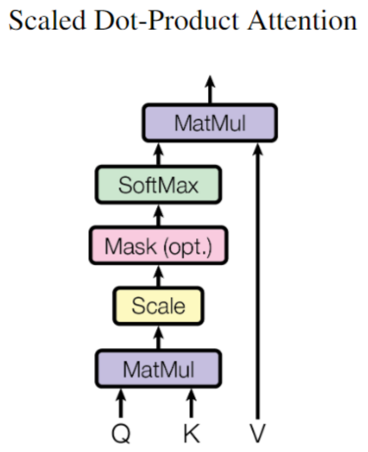


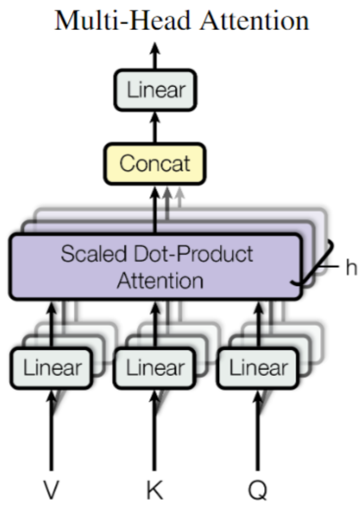
Foundational Challenges in Assuring Alignment and Safety of Large Language Models, arxiv, 2024

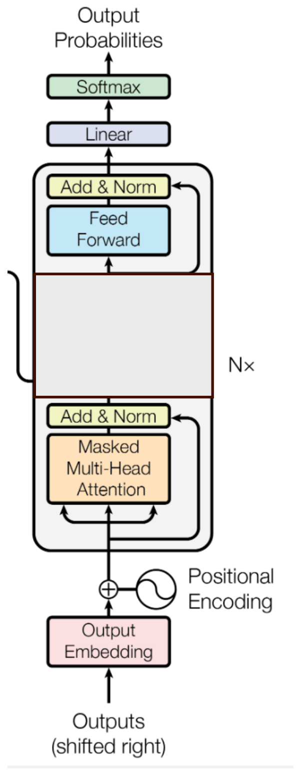
Emergence of ICL

Transformer architecture (Mamba, RWKV…?)

- Transformer (decoder) is the underlying architecture of foundation models.
  - Parallel computing
  - Long distance modeling
  - Unifying different modals…





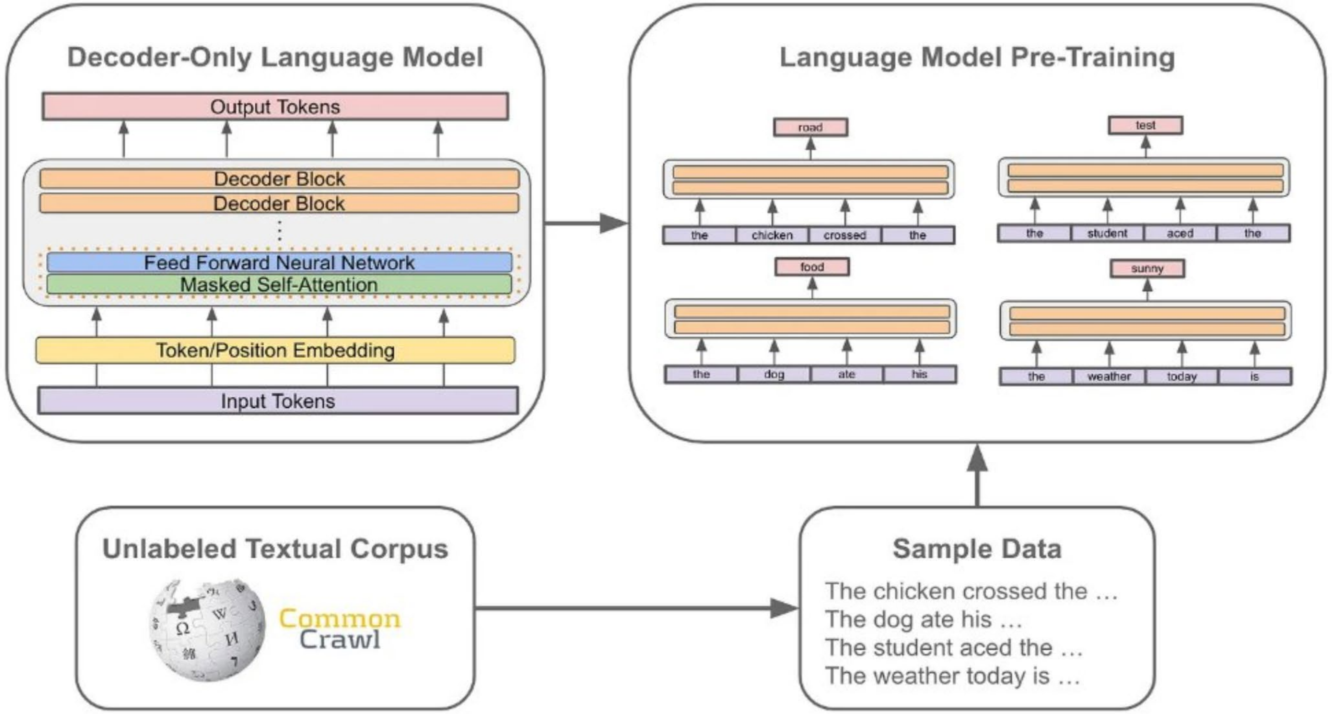


Attention Is All You Need, NeurIPS, 2017

Emergence of ICL

Autoregressive (AR) pretraining

- AR pretraining (next-token prediction) is a simple yet profound SSL method.
  - maximum likelihood training, or minimizing KL(data distribution||model distribution).



Language Models are Few-Shot Learners, NeurIPS, 2020

Emergence of ICL

Model Warmup (optional)

- Warmup adjusts pretrained foundation models before ICL inference.
  - It does not aim the specific tasks but enhances the overall ICL capability of the model.
  - We only focus on MetaICL in this talk, though instruction/symbol tuning… are more popular.

	Meta-training	Inference
Task	C *meta-training* tasks	An unseen *target* task
Data given	Training examples $\mathcal{T}_i = \{(x_j^i, y_j^i)\}_{j=1}^{N_i},\ \forall i \in [1, C] \quad (N_i \gg k)$	Training examples $(x_1, y_1), \cdots, (x_k, y_k)$, Test input x
Objective	For each iteration, 1. Sample task $i \in [1, C]$ 2. Sample $k+1$ examples from $\mathcal{T}_i$: $(x_1, y_1), \cdots, (x_{k+1}, y_{k+1})$ 3. Maximize $P(y_{k+1} \vert x_1, y_1, \cdots, x_k, y_k, x_{k+1})$	$\text{argmax}_{c \in \mathcal{C}} P(c \vert x_1, y_1, \cdots, x_k, y_k, x)$

MetaICL: Learning to Learn In Context, ACL, 2022

Mechanisms of ICL?

Mesa-optimization hypothesis

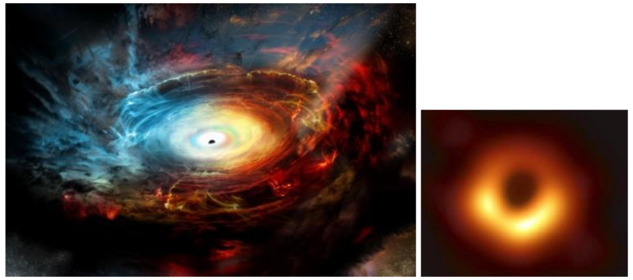
- Hypothesis: the forward pass of the trained transformer is equivalent to optimizing an inner objective function in-context: length generalization?

- How to study this? What is the methodology of fundamental research?
  - Conduct empirical study and summarize common phenomena.
  - Establish theory to interpret these phenomena.

Methodology

Empirical study & theory

Physics



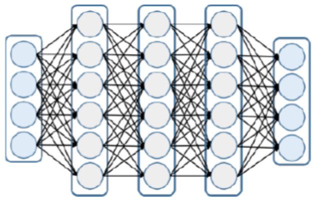
physical phenomenon

$$R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R = \frac{8\pi G}{c^4} T_{\mu\nu}$$

- Theory of Relative
  - Riemannian geometry
- Quantum mechanics
  - Functional analysis

Mathematics

Machine learning



Deep learning

Several mathematicians/physicists join the ML community.

- Prob. theory
- Functional anal.
- Wasserstein geom.
- Diffusion equation
- Statistics
- Optimization
- Numerical analysis

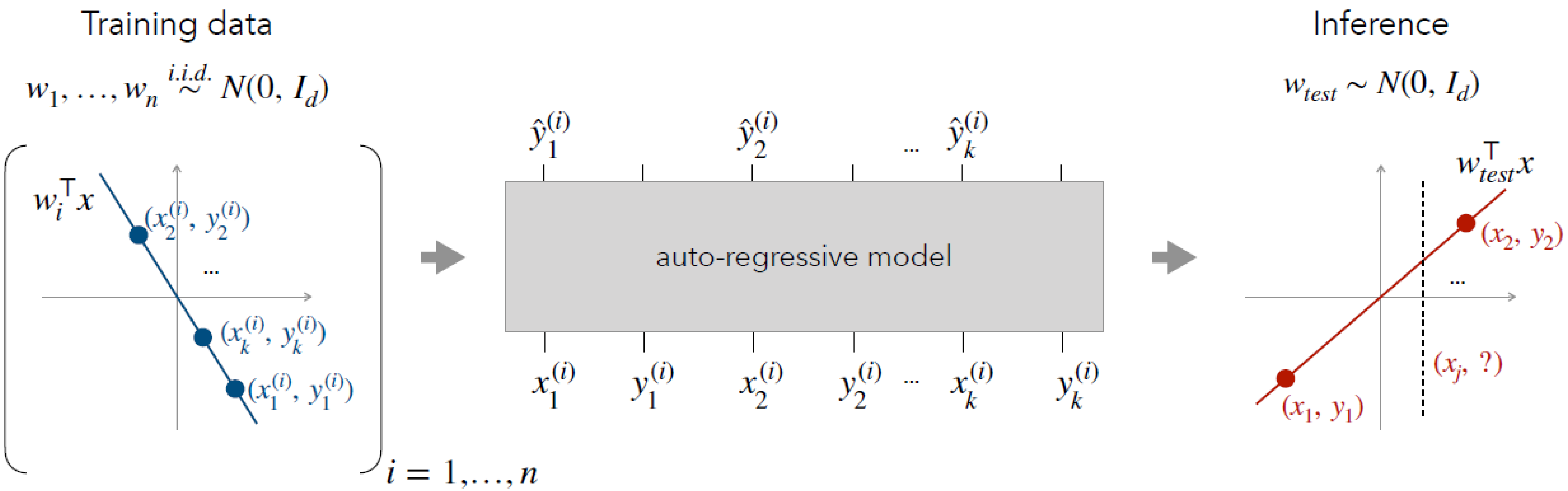
Mathematics

Table of Contents

Empirical findings

A case study on linear functions

- We train a transformer (GPT-2) with (AR) MetaICL objective from scratch.

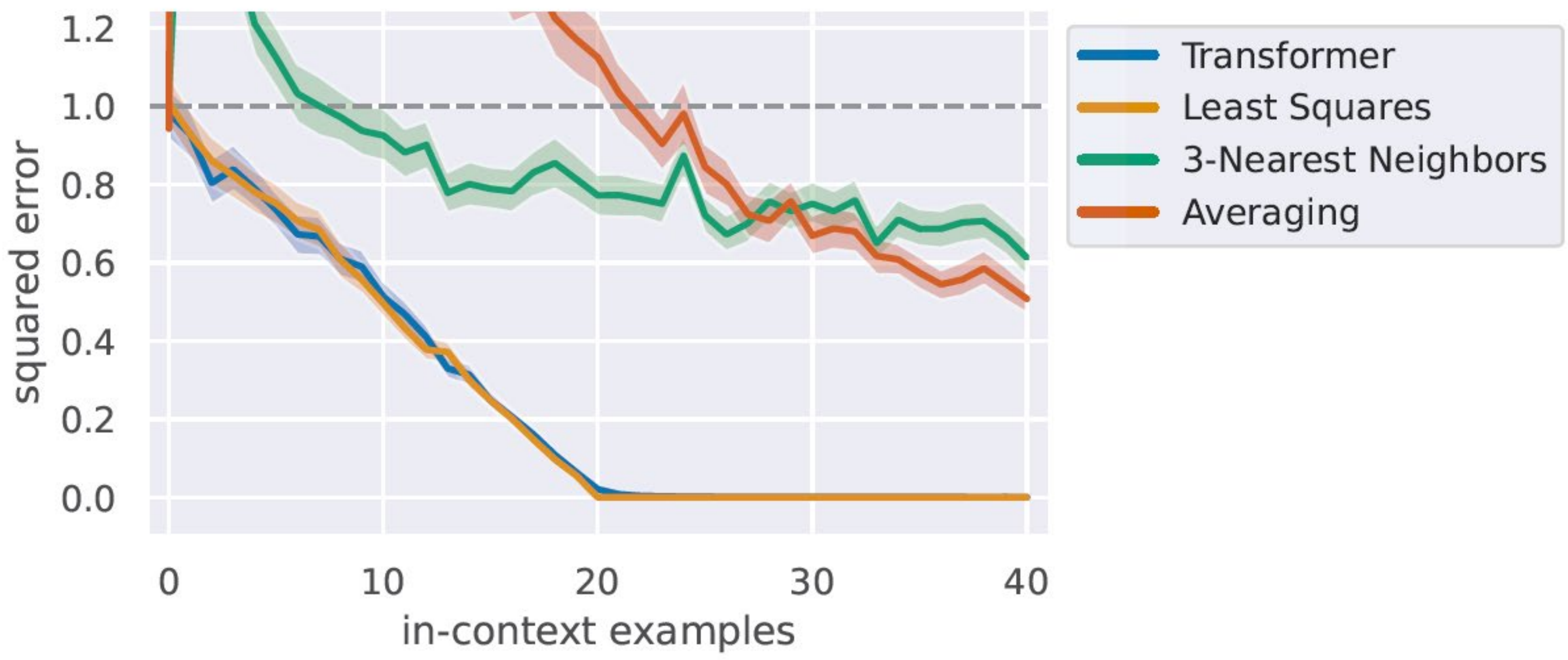


What Can Transformers Learn In-Context? A Case Study of Simple Function Classes, NeurIPS, 2022

Empirical findings

A case study on linear functions

- Given clean test prompt, transformer closely matches the optimal least squares estimator.

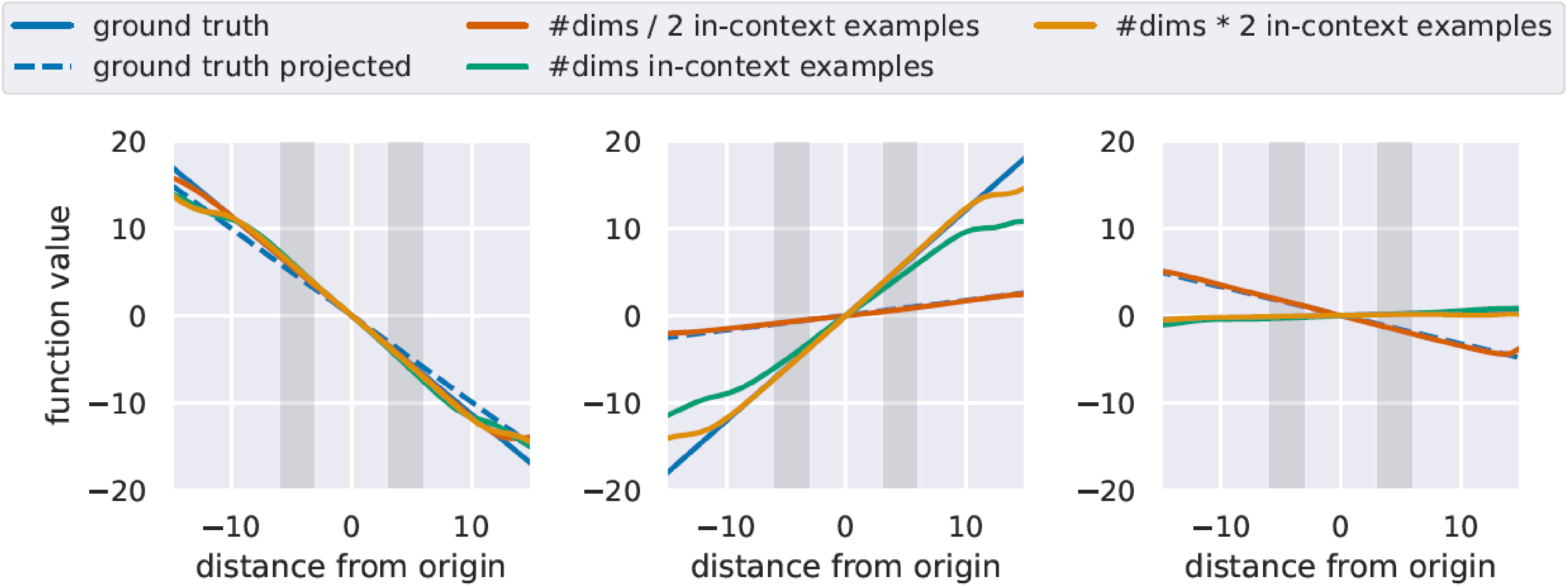


What Can Transformers Learn In-Context? A Case Study of Simple Function Classes, NeurIPS, 2022

Empirical findings

A case study on linear functions

- Given clean test prompt, transformer closely matches least squares estimator.



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

Empirical findings

A case study on linear functions

- Is trained transformer really the same as LSE?: further try OOD settings.

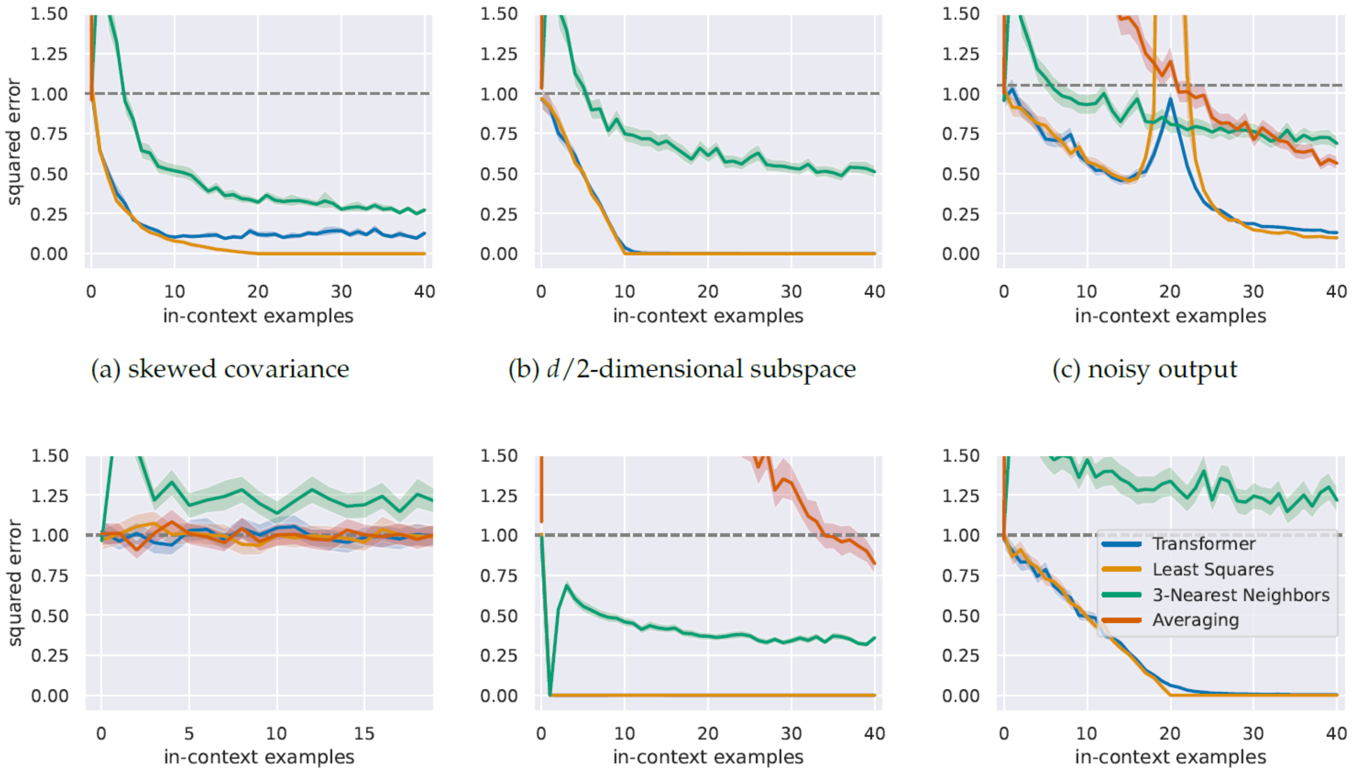
Prompting strategy	$D_{\mathcal{X}}^{\text{train}} \neq D_{\mathcal{X}}^{\text{test}}$	$D_{\mathcal{F}}^{\text{train}} \neq D_{\mathcal{F}}^{\text{test}}$	$D_{\text{query}}^{\text{test}} \neq D_{\mathcal{X}}^{\text{test}}$
Skewed covariance	✓		
$d/2$-dimensional subspace	✓		
Scale inputs	✓		
Noisy output		✓	
Scale weights		✓	
Different Orthants	✓		✓
Orthogonal query			✓
Query matches example			✓

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

Empirical findings

A case study on linear functions

- Trained transformer is not exact LSE, but robust to some distribution shifts.



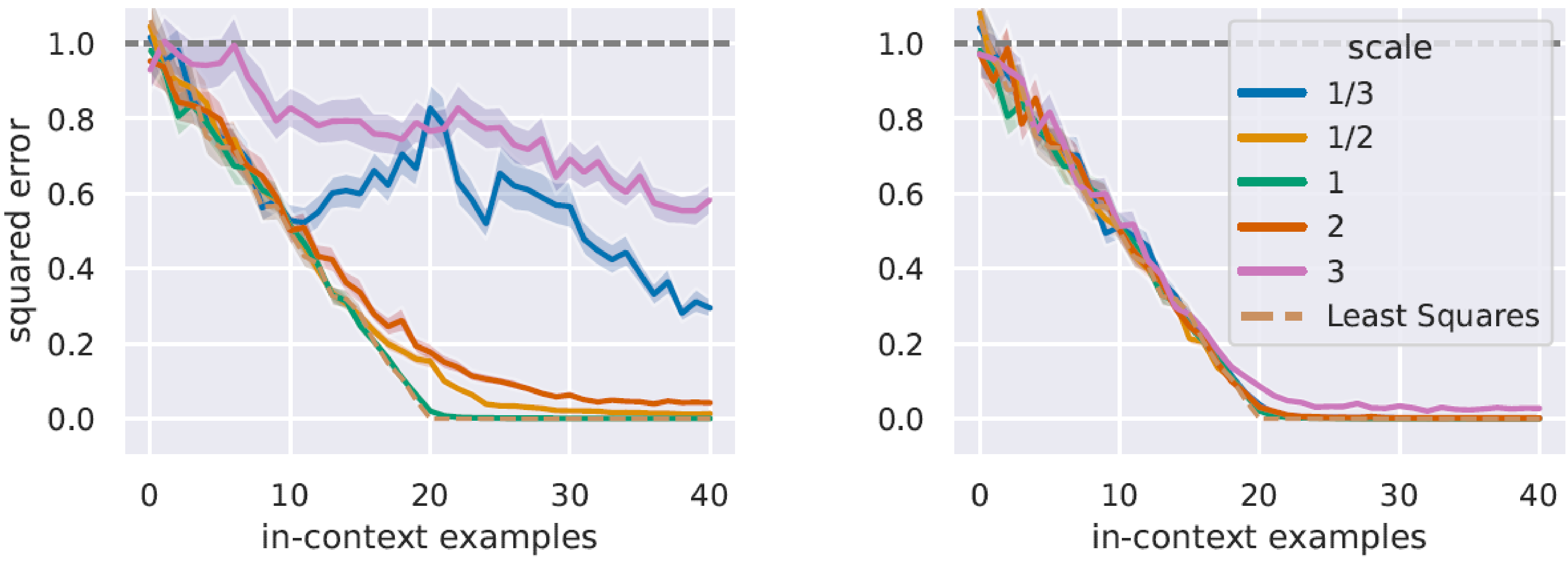
(a) skewed covariance (b) $d/2$-dimensional subspace (c) noisy output
(d) orthogonal query (e) query matches in-context example (f) different orthants

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

Empirical findings

A case study on linear functions

- Trained transformer is not exact LSE, but robust to some distribution shifts.



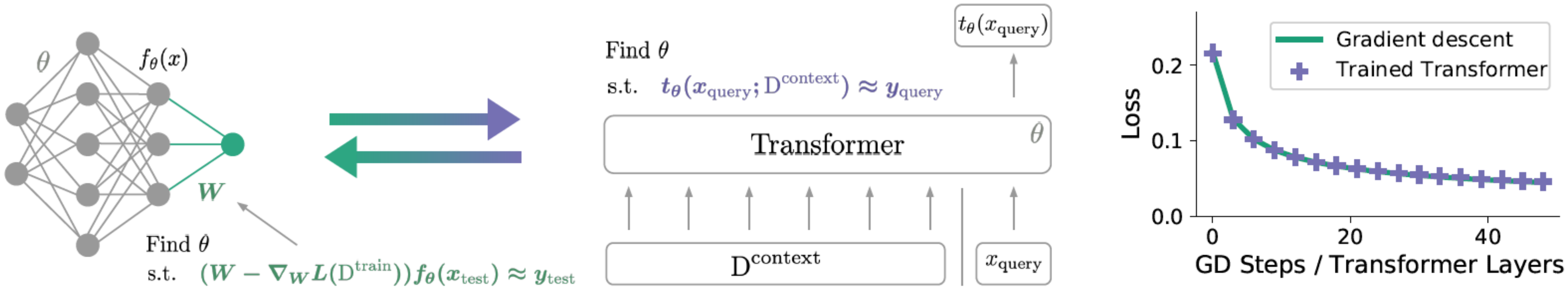
(a) scaled x, Transformer　　(b) scaled w, Transfomer

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

Mesa-optimization hypothesis

Transformers perform gradient descent to approximate LSE?

- Details of experiments is placed in next subsection.
- Hypothesis: trained transformers perform GD to minimize some inner objective in-context.



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

Mesa-optimization hypothesis

Transformers perform gradient descent to approximate LSE?

- Theoretically, with suitable embeddings, the forward pass of one-layer linear attention can express one step of GD on the OLS problem over the context

$L(W) = \frac{1}{2N}\sum_{i=1}^{N} \|Wx_i - y_i\|^2$ with learning rate η which yields weight change

$$\Delta W = -\eta \nabla_W L(W) = -\frac{\eta}{N}\sum_{i=1}^{N}(Wx_i - y_i)x_i^T.$$

$$\begin{pmatrix} x_j \\ y_j \end{pmatrix} \leftarrow \begin{pmatrix} x_j \\ y_j \end{pmatrix} + \frac{\eta}{N} I \sum_{i=1}^{N} \left(\begin{pmatrix} 0 & 0 \\ W_0 & -I_y \end{pmatrix} \begin{pmatrix} x_i \\ y_i \end{pmatrix} \right) \otimes \left(\begin{pmatrix} I_x & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \end{pmatrix} \right) \begin{pmatrix} I_x & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_j \\ y_j \end{pmatrix}$$

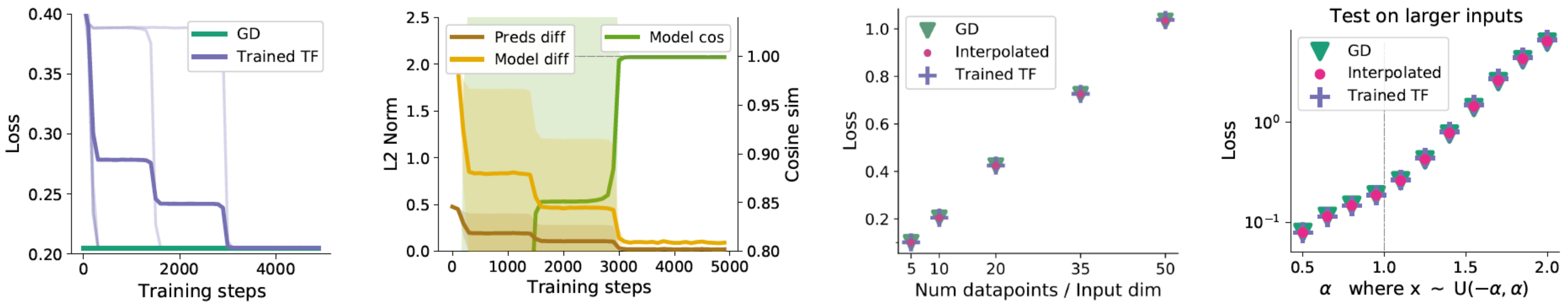
$$= \begin{pmatrix} x_j \\ y_j \end{pmatrix} + \frac{\eta}{N} I \sum_{i=1}^{N} \begin{pmatrix} 0 \\ W_0 x_i - y_i \end{pmatrix} \otimes \begin{pmatrix} x_i \\ 0 \end{pmatrix} \begin{pmatrix} x_j \\ 0 \end{pmatrix} = \begin{pmatrix} x_j \\ y_j \end{pmatrix} + \begin{pmatrix} 0 \\ -\Delta W x_j \end{pmatrix}.$$

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

Mesa-optimization hypothesis

Transformers perform gradient descent to approximate LSE?

- Empirically, the forward pass of trained one-layer linear attention can be captured by one step of GD on the OLS problem over the context, even in OOD setting.



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

Theoretical results

Meta-Trained Transformers is a mesa-optimizer

- Architecture: One-layer linear self-attention module with residual.

$$f_{\mathsf{Attn}}(E; W^K, W^Q, W^V, W^P) = E + W^P W^V E \cdot \mathrm{softmax}\left(\frac{(W^K E)^\top W^Q E}{\rho}\right)$$

Drop out softmax operator

$$f_{\mathsf{LSA}}(E;\theta) = E + W^{PV} E \cdot \frac{E^\top W^{KQ} E}{\rho}$$

Trained Transformers Learn Linear Models In-Context, JMLR, 2024

Theoretical results on Meta ICL

Meta-Trained Transformers is a mesa-optimizer

- Data: $x_i, x_{query} \sim N(0, \Lambda)$ and $w \sim N(0, I_d)$.
- Embeddings (important): stack historical linear problem examples, and add the query input with 0 left for storing the prediction result.

$$E_\tau := \begin{pmatrix} x_{\tau,1} & x_{\tau,2} & \cdots & x_{\tau,N} & x_{\tau,\text{query}} \\ \langle w_\tau, x_{\tau,1} \rangle & \langle w_\tau, x_{\tau,2} \rangle & \cdots & \langle w_\tau, x_{\tau,N} \rangle & 0 \end{pmatrix}$$

Trained Transformers Learn Linear Models In-Context, JMLR, 2024

Theoretical results on Meta ICL

Meta-Trained Transformers is a mesa-optimizer

- Population loss.

$$L(\theta) = \lim_{B \to \infty} \widehat{L}(\theta) = \frac{1}{2}\mathbb{E}_{w_\tau, x_{\tau,1}, \cdots, x_{\tau,N}, x_{\tau,\text{query}}} \left[(\widehat{y}_{\tau,\text{query}} - \langle w_\tau, x_{\tau,\text{query}} \rangle)^2 \right]$$

- We use gradient flow to optimize the loss function.

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta = -\nabla L(\theta).$$

Trained Transformers Learn Linear Models In-Context, JMLR, 2024

Theoretical results on Meta ICL

Meta-Trained Transformers is a mesa-optimizer

- Initialization. We let the zero matrices in the before theoretical construction as zero at the initial time.

Assumption 3.3 (Initialization). *Let $\sigma > 0$ be a parameter, and let $\Theta \in \mathbb{R}^{d\times d}$ be any matrix satisfying $\|\Theta\Theta^\top\|_F = 1$ and $\Theta\Lambda \neq 0_{d\times d}$. We assume*

$$W^{PV}(0) = \sigma \begin{pmatrix} 0_{d\times d} & 0_d \\ 0_d^\top & 1 \end{pmatrix}, \quad W^{KQ}(0) = \sigma \begin{pmatrix} \Theta\Theta^\top & 0_d \\ 0_d^\top & 0 \end{pmatrix}. \tag{3.10}$$

Trained Transformers Learn Linear Models In-Context, JMLR, 2024

Theoretical results on Meta ICL

Meta-Trained Transformers is a mesa-optimizer

- Convergence results. Let $\Gamma = \left(1 + \frac{1}{N}\right)\Lambda + \frac{1}{N} tr(\Lambda) I_d$, we have

Then gradient flow converges to a global minimum of the population loss (3.8). Moreover, W^{PV} and W^{KQ} converge to W_*^{PV} and W_*^{KQ} respectively, where

$$W_*^{KQ} = \left[\mathrm{tr}\left(\Gamma^{-2}\right)\right]^{-\frac{1}{4}} \begin{pmatrix} \Gamma^{-1} & 0_d \\ 0_d^\top & 0 \end{pmatrix}, \qquad W_*^{PV} = \left[\mathrm{tr}\left(\Gamma^{-2}\right)\right]^{\frac{1}{4}} \begin{pmatrix} 0_{d\times d} & 0_d \\ 0_d^\top & 1 \end{pmatrix}. \tag{4.1}$$

- When $\Lambda = \sigma^2 I_d$, then $\Gamma = \left(1 + \frac{d+1}{N}\right)\sigma^2 I_d$, which exactly matches the theoretical construction to perform one step of GD.

Trained Transformers Learn Linear Models In-Context, JMLR, 2024

Theoretical results on Meta ICL

Meta-Trained Transformers is a mesa-optimizer

- In general, trained transformer implement one step of preconditioned GD, and optimally solve the linear regression task with long enough prompts.
- It is not LSE yet.

$$
\begin{aligned}
\widehat{y}_{\text{query}} &= \begin{pmatrix} 0_d^\top & 1 \end{pmatrix} \begin{pmatrix} \frac{1}{M}\sum_{i=1}^M x_i x_i^\top + \frac{1}{M} x_{\text{query}} x_{\text{query}}^\top & \frac{1}{M}\sum_{i=1}^M x_i x_i^\top w \\ \frac{1}{M}\sum_{i=1}^M w^\top x_i x_i^\top & \frac{1}{M}\sum_{i=1}^M w^\top x_i x_i^\top w \end{pmatrix} \begin{pmatrix} \Gamma^{-1} & 0_d \\ 0_d^\top & 0 \end{pmatrix} \begin{pmatrix} x_{\text{query}} \\ 0 \end{pmatrix} \\
&= x_{\text{query}}^\top \Gamma^{-1} \boxed{\left(\frac{1}{M} \sum_{i=1}^M x_i x_i^\top \right) w.} \qquad (4.2)
\end{aligned}
$$

One step size preconditioned GD for the OLS problem over context!

When the length of prompts seen during training N is large, $\Gamma^{-1} \approx \Lambda^{-1}$, and when the test prompt length M is large, $\frac{1}{M}\sum_{i=1}^M x_i x_i^\top \approx \Lambda$, so that $\widehat{y}_{\text{query}} \approx x_{\text{query}}^\top w$. Thus, for sufficiently large prompt lengths, *the trained transformer indeed in-context learns the class of linear predictors.*

Trained Transformers Learn Linear Models In-Context, JMLR, 2024

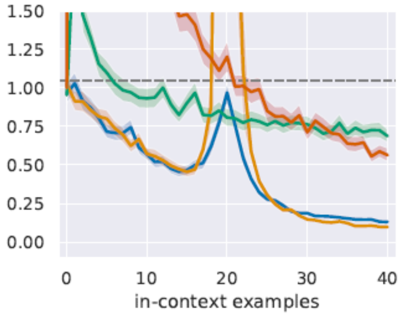
Theoretical results on Meta ICL

Meta-Trained Transformers is a mesa-optimizer

- Trained transformer is robust to task shifts.

For example, consider a prompt corresponding to a noisy linear model, so that the prompt consists of a sequence of (x_i, y_i) pairs where $y_i = \langle w, x_i \rangle + \varepsilon_i$ for some arbitrary vector $w \in \mathbb{R}^d$ and independent sub-Gaussian noise ε_i. Then from (4.7), the prediction of the transformer on query examples is

$$\widehat{y}_{\text{query}} \approx x_{\text{query}}^\top \Lambda^{-1} \left(\frac{1}{M} \sum_{i=1}^{M} y_i x_i \right) = x_{\text{query}}^\top \Lambda^{-1} \left(\frac{1}{M} \sum_{i=1}^{M} x_i x_i^\top \right) w + x_{\text{query}}^\top \Lambda^{-1} \left(\frac{1}{M} \sum_{i=1}^{M} \varepsilon_i x_i \right).$$



(b) Noisy linear regression

Trained Transformers Learn Linear Models In-Context, JMLR, 2024

Theoretical results on Meta ICL

Meta-Trained Transformers is a mesa-optimizer

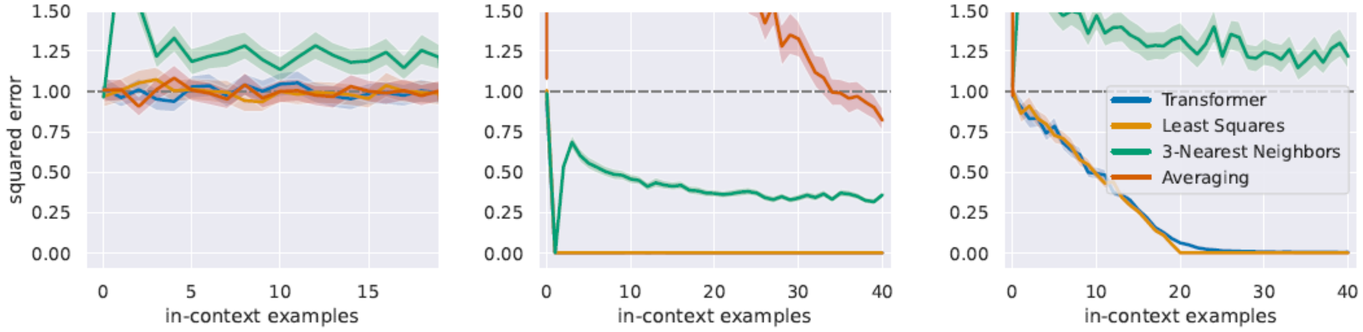
- Trained transformer is robust to query shifts.

Query shifts. Continuing from (4.7), since $y_i = \langle w, x_i \rangle$,

$$\widehat{y}_{\text{query}} \approx x_{\text{query}}^\top \Lambda^{-1} \left(\frac{1}{M} \sum_{i=1}^{M} x_i x_i^\top \right) w.$$

From this we see that whether query shifts can be tolerated hinges upon the distribution of the x_i's. Since $\mathcal{D}_x^{\text{train}} = \mathcal{D}_x^{\text{test}}$, if M is large then

$$\widehat{y}_{\text{query}} \approx x_{\text{query}}^\top \Lambda^{-1} \Lambda w = x_{\text{query}}^\top w. \tag{4.8}$$



(d) orthogonal query (e) query matches in-context example (f) different orthants

Trained Transformers Learn Linear Models In-Context, JMLR, 2024

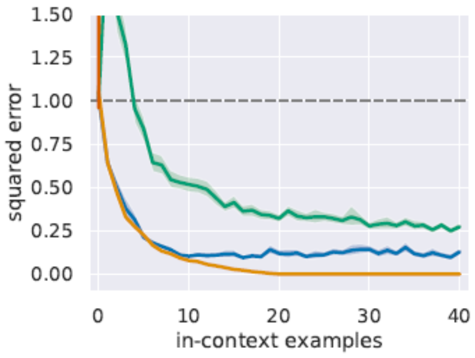
Theoretical results on Meta ICL

Meta-Trained Transformers is a mesa-optimizer

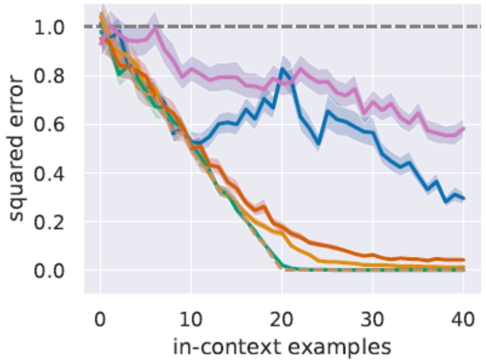
- Trained transformer is not robust to query shifts.

Covariate shifts. In contrast to task and query shifts, covariate shifts cannot be fully tolerated in the transformer. This can be easily seen due to the identity (4.3): when $\mathcal{D}_x^{\text{train}} \neq \mathcal{D}_x^{\text{test}}$, then the approximation in (4.8) does not hold as $\frac{1}{M}\sum_{i=1}^{M} x_i x_i^\top$ will not cancel Γ^{-1} when M and N are large. For instance, if we consider test prompts where the covariates are scaled by a constant $c \neq 1$, then

$$\widehat{y}_{\text{query}} \approx x_{\text{query}}^\top \Lambda^{-1} \left(\frac{1}{M} \sum_{i=1}^{M} x_i x_i^\top \right) \approx x_{\text{query}}^\top \Lambda^{-1} c^2 \Lambda w = c^2 x_{\text{query}}^\top w \neq x_{\text{query}}^\top w.$$



(a) skewed covariance



(a) scaled x, Transformer

Trained Transformers Learn Linear Models In-Context, JMLR, 2024

Summary on Meta ICL

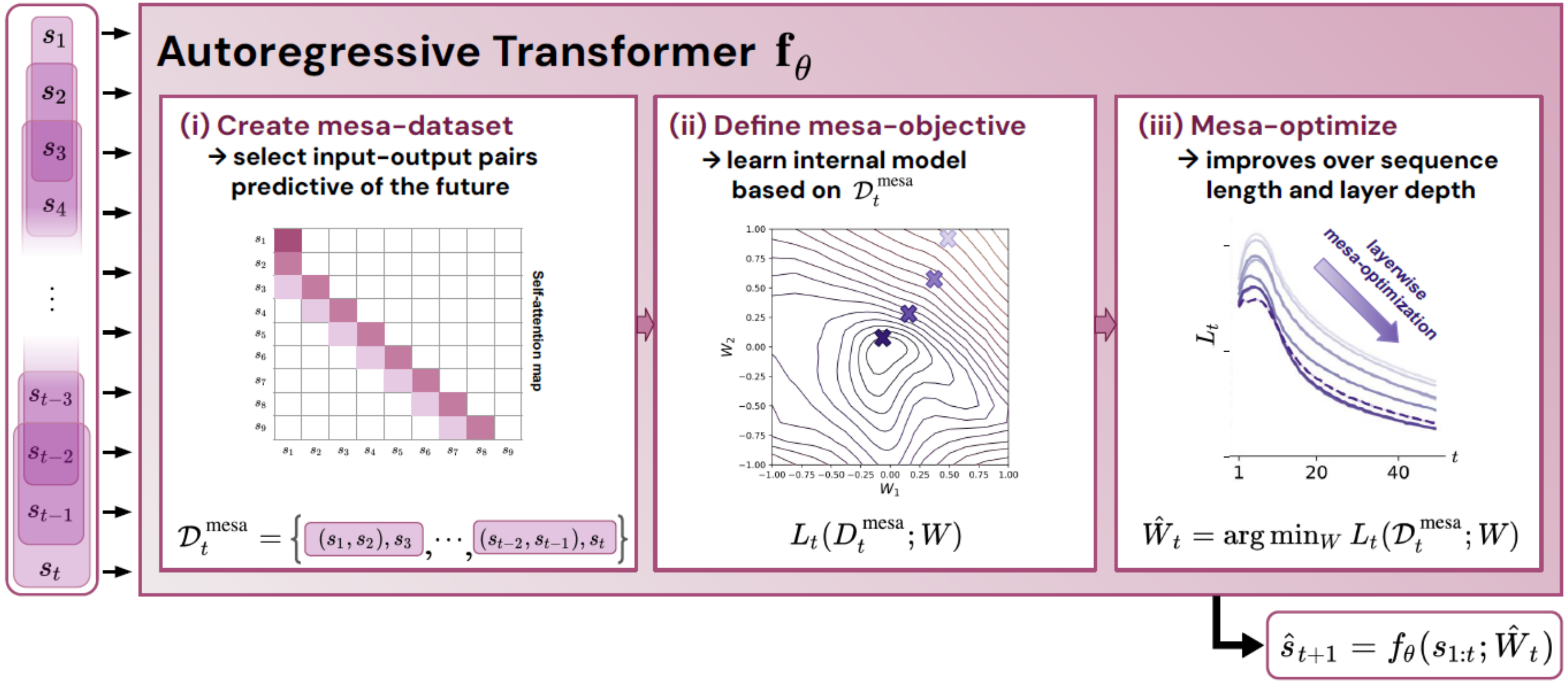
- Empirical findings: practical transformer closely matches the optimal LSE.
- Mesa-optimization hypothesis: trained transformer perform GD-based algorithm.
  - A theoretical construction without optimization guarantee.
  - Empirical evidence on one-layer linear attention.
- Non-trivial theoretical framework based on feature learning theory.
  - Trained one-layer linear attention do implement GD!
  - Interpret the practical transformer in OOD settings.

Table of Contents

Mesa-optimization hypothesis

Transformers perform gradient descent to approximate LSE?

- Hypothesis: Autoregressively trained transformers also perform GD to minimize some inner objective in-context.

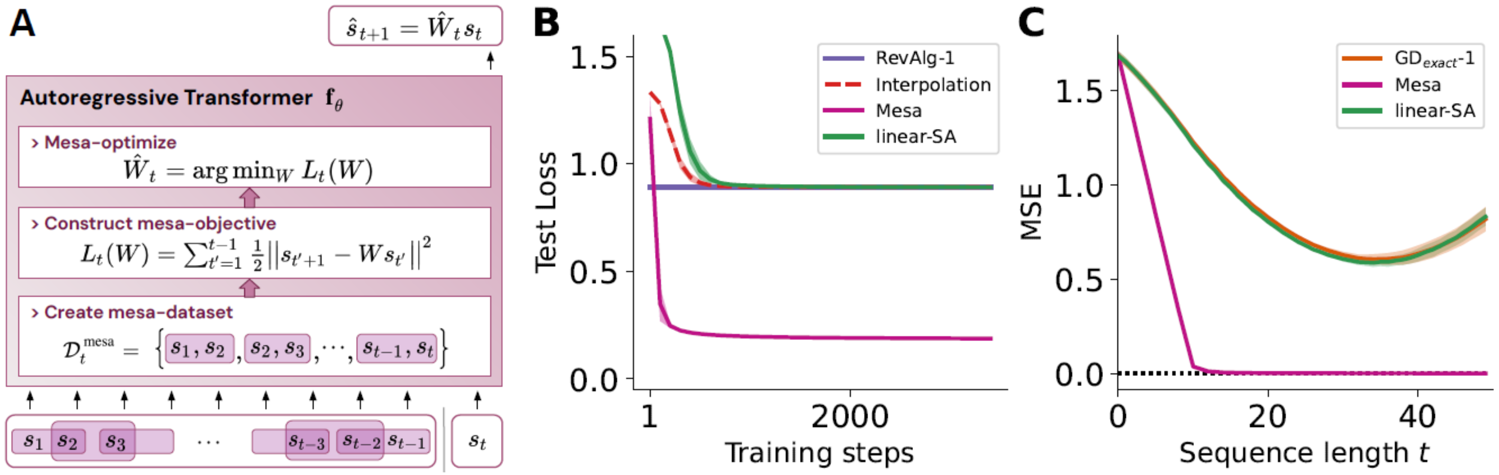


Uncovering mesa-optimization algorithms in Transformers, ICLR-W, 2024

Mesa-optimization hypothesis

Transformers perform gradient descent to approximate LSE?

- Theoretically, with suitable embeddings, the forward pass of one-layer linear attention can express one step of GD on the OLS problem over the context.
- Empirically, the forward pass of autoregressively trained one-layer linear attention can be captured by one step of GD.



Uncovering mesa-optimization algorithms in Transformers, ICLR-W, 2024

Theoretical analyses

Mesa-optimization in Autoregressively trained transformers

On Mesa-Optimization in Autoregressively Trained Transformers: Emergence and Capability

Chenyu Zheng[1], Wei Huang[2], Rongzhen Wang[1], Guoqiang Wu[3],
Jun Zhu[4], Chongxuan Li[1*]
[1] Gaoling School of Artificial Intelligence, Renmin University of China
[2] RIKEN AIP [3] School of Software, Shandong University
[4] Dept. of Comp. Sci. & Tech., BNRist Center, THU-Bosch ML Center, Tsinghua University
{cyzheng,wangrz,chongxuanli}@ruc.edu.cn; wei.huang.vr@riken.jp;
guoqiangwu@sdu.edu.cn; dcszj@mail.tsinghua.edu.cn

On Mesa-Optimization in Autoregressively Trained Transformers: Emergence and Capability, arxiv, 2024

Theoretical analyses

Sequence distribution

- Initial point $x_1 \sim D_{x_1}$, we discuss its impact on trained transformer.
- 1-st order AR process: $x_{t+1} = Wx_t$.
- W is uniformly sampled from diagonal unitary complex matrix.

On Mesa-Optimization in Autoregressively Trained Transformers: Emergence and Capability, arxiv, 2024

Theoretical analyses

Architecture

- Architecture: One-layer linear causal self-attention module with residual.

$$f_t(\boldsymbol{E}_t;\boldsymbol{\theta}) = \boldsymbol{e}_t + \boldsymbol{W}^{PV}\boldsymbol{E}_t \cdot \frac{\boldsymbol{E}_t^*\boldsymbol{W}^{KQ}\boldsymbol{e}_t}{\rho_t}.$$

- Embeddings: natural extension of that in MetaICL setting. 0 is left for storing prediction results.

$$\boldsymbol{E}_t = (\boldsymbol{e}_1,\ldots,\boldsymbol{e}_t) = \begin{pmatrix} \boldsymbol{0}_d & \boldsymbol{0}_d & \cdots & \boldsymbol{0}_d \\ \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_t \\ \boldsymbol{x}_0 & \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_{t-1} \end{pmatrix}$$

On Mesa-Optimization in Autoregressively Trained Transformers: Emergence and Capability, arxiv, 2024

Theoretical analyses

Loss function and initialization

- Loss function: we use gradient flow on next-token prediction loss.

$$L(\boldsymbol{\theta}) = \sum_{t=2}^{T-1} L_t(\boldsymbol{\theta}) = \sum_{t=2}^{T-1} \mathbb{E}_{\boldsymbol{x}_1, \boldsymbol{W}} \left[\frac{1}{2} \|\widehat{\boldsymbol{y}}_t - \boldsymbol{x}_{t+1}\|_2^2 \right],$$

- Initialization: we let the zero matrices in the ideal theoretical construction as zero at the initial time.

Assumption 3.1 (Initialization). *At the initial time $\tau = 0$, we assume that*

$$\boldsymbol{W}^{KQ}(0) = \begin{pmatrix} \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} \\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} \\ \mathbf{0}_{d\times d} & a_0 \boldsymbol{I}_d & \mathbf{0}_{d\times d} \end{pmatrix}, \boldsymbol{W}^{PV}(0) = \begin{pmatrix} \mathbf{0}_{d\times d} & b_0 \boldsymbol{I}_d & \mathbf{0}_{d\times d} \\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} \\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} \end{pmatrix}$$

On Mesa-Optimization in Autoregressively Trained Transformers: Emergence and Capability, arxiv, 2024

Theoretical analyses

Existing results

- $x_1 = 1_d$.
- Red matrices are all diagonal.
- Only focus on the property of global minima, without convergence guarantee.

Proposition 2 (In-context autoregressive learning with gradient-descent). *Suppose assumptions 1 and 2. Loss (2) is minimal for $a_1 + a_4 = b_2 = 0$ and $a_3 b_1 = \frac{\sum_{T=2}^{T_{\max}} T}{\sum_{T=2}^{T_{\max}} (T^2 + (d-1)T)}$. Furthermore, the optimal in-context map Γ_{θ^*} is one step of gradient descent starting from the initialization $\lambda = 0$, with a step size asymptotically equivalent to $\frac{3}{2T_{\max}}$ with respect to $T_{\max}$.*

How do Transformers perform In-Context Autoregressive Learning?, ICML, 2024

Theoretical analyses

When does dynamics converge to ideal theoretical construction?

Assumption 4.1 (Sufficient condition for the emergence of mesa-optimizer). *We assume that the distribution $\mathcal{D}_{\boldsymbol{x}_1}$ of the initial token $\boldsymbol{x}_1 \in \mathbb{R}$ satisfies $\mathbb{E}_{\boldsymbol{x}_1 \sim \mathcal{D}_{\boldsymbol{x}_1}}[x_{1i_1} x_{1i_2}^{r_2} \cdots x_{1i_n}^{r_n}] = 0$ for any subset $\{i_1, \ldots, i_n \mid n \leq 4\}$ of $[d]$, and $r_2, \ldots r_n \in \mathbb{N}$. In addition, we assume that $\kappa_1 = \mathbb{E}[x_{1j}^4]$, $\kappa_2 = \mathbb{E}[x_{1j}^6]$ and $\kappa_3 = \sum_{r \neq j} \mathbb{E}[x_{1j}^2 x_{1r}^4]$ are finite constant for any $j \in [d]$.*

- We note that any random vectors whose coordinates are i.i.d. random variables with zero mean satisfy this assumption, such as $N(0, I_d)$.

On Mesa-Optimization in Autoregressively Trained Transformers: Emergence and Capability, arxiv, 2024

Theoretical analyses

Convergence results

- Autoregressively trained transformer converges to the ideal case.

Theorem 4.1 (Convergence of the gradient flow, proof in Section 5). *Consider the gradient flow of the one-layer linear transformer (see Eq. 1) over the population AR pretraining loss (see Eq. 2). Suppose the initialization satisfies Assumption 3.1, and the initial token's distribution $\mathcal{D}_{\boldsymbol{x}_1}$ satisfies Assumption 4.1, then the gradient flow converges to*

$$\begin{pmatrix} \widetilde{\boldsymbol{W}_{22}^{KQ}} & \widetilde{\boldsymbol{W}_{23}^{KQ}} \\ \widetilde{\boldsymbol{W}_{32}^{KQ}} & \widetilde{\boldsymbol{W}_{33}^{KQ}} \end{pmatrix} = \begin{pmatrix} \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} \\ \widetilde{a}\boldsymbol{I}_d & \mathbf{0}_{d\times d} \end{pmatrix}, \begin{pmatrix} \widetilde{\boldsymbol{W}_{12}^{PV}} & \widetilde{\boldsymbol{W}_{13}^{PV}} \end{pmatrix} = \begin{pmatrix} \widetilde{b}\boldsymbol{I}_d & \mathbf{0}_{d\times d} \end{pmatrix}.$$

Though different initialization (a_0, b_0) lead to different $(\widetilde{a}, \widetilde{b})$, the solutions' product $\widetilde{a}\widetilde{b}$ satisfies

$$\widetilde{a}\widetilde{b} = \frac{\kappa_1}{\kappa_2 + \frac{\kappa_3}{T-2}\sum_{t=2}^{T-1}\frac{1}{t-1}}.$$

On Mesa-Optimization in Autoregressively Trained Transformers: Emergence and Capability, arxiv, 2024

Theoretical analyses

Convergence results

- Autoregressively trained transformer implements GD for the OLS problem.

Corollary 4.1 (Trained transformer as a mesa-optimizer, proof in Appendix A.3). *We suppose that the same precondition of Theorem 4.1 holds. When predicting the* $(t+1)$*-th token, the trained transformer obtains* $\widehat{\mathbf{W}}$ *by implementing one step of gradient descent for the OLS problem* $L_{\mathrm{OLS},t}(\boldsymbol{W}) = \frac{1}{2}\sum_{i=1}^{t-1}\|\boldsymbol{x}_{i+1} - W\boldsymbol{x}_i\|^2$*, starting from the initialization* $\boldsymbol{W} = \mathbf{0}_{d\times d}$ *with a step size* $\frac{\widetilde{ab}}{t-1}$.

On Mesa-Optimization in Autoregressively Trained Transformers: Emergence and Capability, arxiv, 2024

Theoretical analyses

Capability of Mesa-optimizer

- Mesa-optimizer fails to recover process with normal initial token.

$$\widehat{\boldsymbol{y}}_{T_{te}} = \boldsymbol{W}\left(\widetilde{a}\widetilde{b}\frac{\sum_{i=1}^{T_{te}-1}\boldsymbol{x}_i\boldsymbol{x}_i^*}{T_{te}-1}\right)\boldsymbol{x}_{T_{te}}$$

Proposition 4.1 (AR process with normal distributed initial token can not be learned, proof in Appendix A.4). *Let $\mathcal{D}_{\boldsymbol{x}_1}$ be the multivariate normal distribution $\mathcal{N}(\mathbf{0}_d, \sigma^2\boldsymbol{I}_d)$ with any $\sigma^2 \geq 0$, then the "simple" AR process can not be recovered by the trained transformer even in the ideal case with long enough context. Formally, when the training sequence length T_{tr} and test context length T_{te} are large enough, the prediction from the trained transformer satisfies*

$$\mathbb{E}_{\boldsymbol{x}_1}\left[\widetilde{a}\widetilde{b}\frac{\sum_{i=1}^{T_{te}-1}\boldsymbol{x}_i\boldsymbol{x}_i^*}{T_{te}-1}\right] \to \frac{1}{5}\boldsymbol{I}_d, \quad T_{tr}, T_{te} \to +\infty.$$

On Mesa-Optimization in Autoregressively Trained Transformers: Emergence and Capability, arxiv, 2024

Theoretical analyses

When does mesa-optimizer recover sequence?

- The sufficient and necessary condition for learning the true distribution.

Assumption 4.2 (Condition for success of mesa-optimizer)**.** *Based on Assumption 4.1, we further suppose that* $\frac{\kappa_1}{\kappa_2}\frac{\sum_{i=1}^{T_{te}-1}\boldsymbol{x}_i\boldsymbol{x}_i^*}{T_{te}-1}\boldsymbol{x}_{T_{te}} \to \boldsymbol{x}_{T_{te}}$ *for any* $\boldsymbol{x}_1$ *and* $\boldsymbol{W}$*, when* T_{te} *is large enough.*

- A toy example that satisfies the assumption.

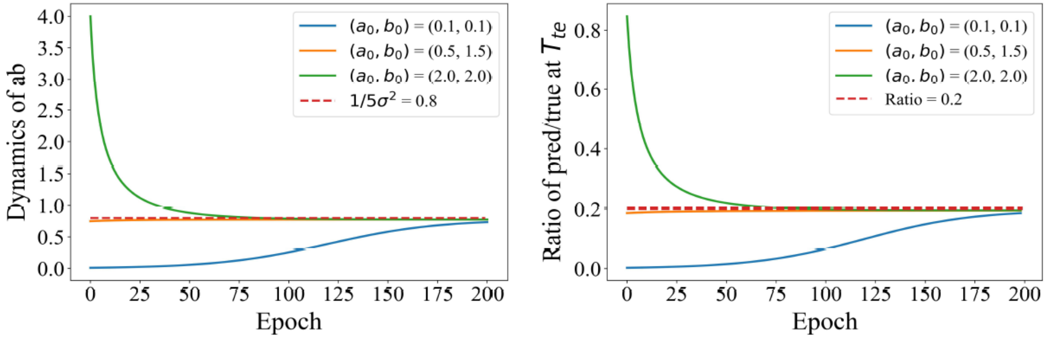
Example 4.1 (sparse vector). If the random vector $\boldsymbol{x}_1 \in R^d$ is uniformly sampled from the candidate set of size $2d$ $\{\pm(c, 0, \ldots, 0)^\top, \pm(0, c, \ldots, 0)^\top, \pm(0, \ldots, 0, c)^\top\}$ for any fixed $c \in \mathbb{R}$, then the distribution $\mathcal{D}_{\boldsymbol{x}_1}$ satisfies Assumption 4.2. The derivation can be found in Appendix A.5.

On Mesa-Optimization in Autoregressively Trained Transformers: Emergence and Capability, arxiv, 2024

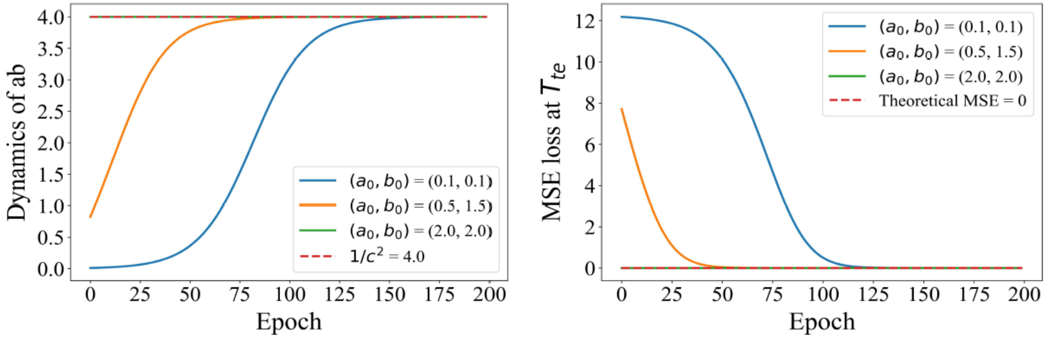
Theoretical analyses

Simulations

- Simulations verify our theoretical results.



(a) Gaussian with $\sigma = 0.5$, dynamics of ab (b) Gaussian with $\sigma = 0.5$, ratio of $\widehat{\boldsymbol{y}}_{T_{te-1}} / \boldsymbol{x}_{T_{te}}$



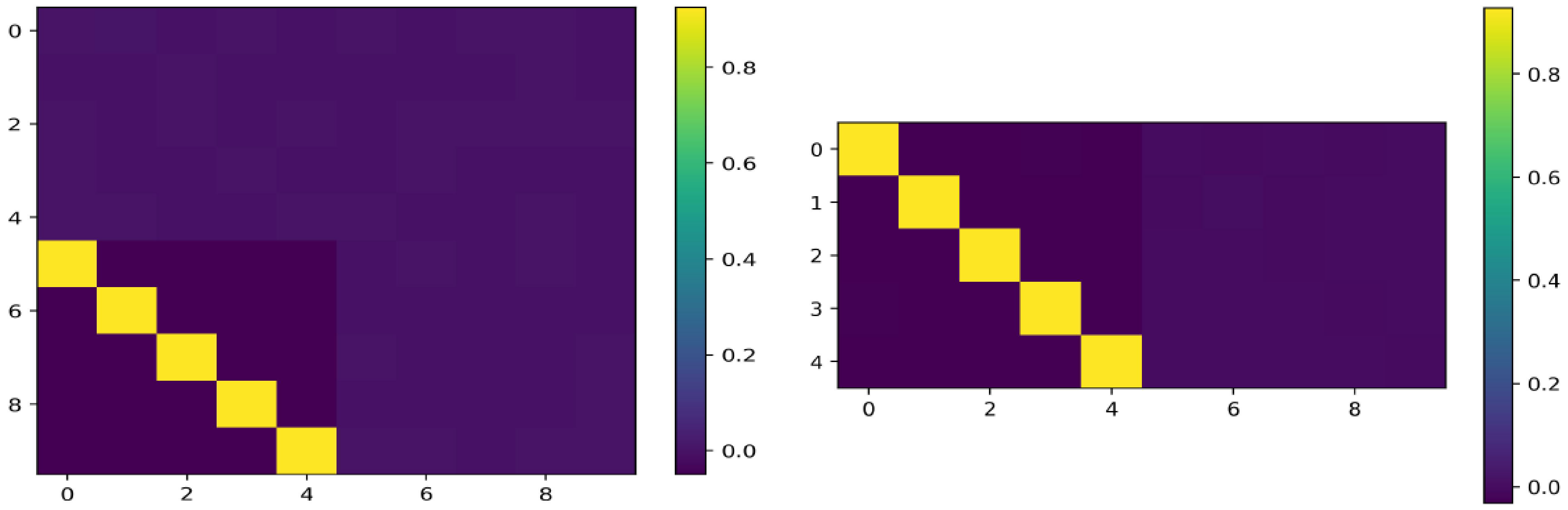
(c) Example 4.1 with $c = 0.5$, dynamics of ab (d) Example 4.1 with $c = 0.5$, $\|\widehat{\boldsymbol{y}}_{T_{te-1}} - \boldsymbol{x}_{T_{te}}\|_2^2$

On Mesa-Optimization in Autoregressively Trained Transformers: Emergence and Capability, arxiv, 2024

Theoretical analyses

Simulations

- We also explore the case beyond the data condition, and suggest it will perform preconditioned GD in general.



(a) $\boldsymbol{W}^{KQ}, a = 0.1, b = 0.1$ (b) $\boldsymbol{W}^{PV}, a = 0.1, b = 0.1$

On Mesa-Optimization in Autoregressively Trained Transformers: Emergence and Capability, arxiv, 2024

Table of Contents

Future works

Both empirical and theoretical directions

- Observe and interpret more complex architectures with different embeddings
  - 1-layer: softmax SA, multi-head SA
  - 2-layer: looped SA, independent LSA, LSA+MLP, Layernorm
  - Transformer block, full transformer
  - Asymptotic theory: infinite width/depth
- Observe and interpret more data settings
  - Noise, imbalanced…
  - Discrete sequence
- Better architecture to perform mesa-optimization
- Relation between AR ICL and Meta ICL