Trained Mamba Emulates Online Gradient Descent in In-Context Linear Regression

Jiarui Jiang*[1], Wei Huang*[2], Miao Zhang†[1], Taiji Suzuki[3,2], Liqiang Nie[1]
[1]Harbin Institute of Technology, Shenzhen, [2]RIKEN AIP, [3]University of Tokyo

NEURAL INFORMATION PROCESSING SYSTEMS

Introduction

What is ICL?

In-context learning (**ICL**) is a powerful paradigm that enables models to generalize to unseen tasks by dynamically leveraging contextual examples (such as input-output pairs) without task-specific fine-tuning.

e.g.

Task: Translate English to French

Input: I love you
Output: Je t'aime

Input: Good morning
Output: Bonjour

Input: How are you?
Output:

prompt (context)

The model needs to learn from the context and then answer this question

Why study ICL on Mamba?

- ICL is a **critical capacity** for large foundation models.
- Mamba is an **efficient** Transformer alternative with linear complexity,
- but **theoretical understanding** of Mamba's ICL remains limited.

Hightlight

- We reveal that trained Mamba can emulate **online gradient descent** in In-Context Linear Regression.
- We establish a **convergence guarantee** for Mamba from random initialization to ICL solution, and further derive the **loss bound** after convergence. The loss matches that of Transformers.

Problem Setup

Data Model

$\boldsymbol{x}_i, \boldsymbol{x}_q, \boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$

For each sequence, we have a new function $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x}$,
Given prompt $\{\boldsymbol{x}_i, y_i\}_{i=1}^N$, where $y_i = f(\boldsymbol{x}_i) = \boldsymbol{w}^\top \boldsymbol{x}_i$,
we expect the model to learn the latent function $f(\cdot)$,
and predict $y_q = f(\boldsymbol{x}_q) = \boldsymbol{w}^\top \boldsymbol{x}_q$ for query $\boldsymbol{x}_q$

$$\begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_N \\ y_1 & y_2 & \cdots & y_N \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_q \\ 0 \end{bmatrix} \quad \text{target: predict } y_q$$

prompt (context)

Mamba Model

Mamba Model. We consider a S6 layer of Mamba $\boldsymbol{o}_{1:L} = \textbf{Mamba}(\boldsymbol{\theta}; \boldsymbol{u}_{1:L})$ with selection, discretization, and linear recurrence components, where $\boldsymbol{u}_l, \boldsymbol{o}_l \in \mathbb{R}^{d_e}$. It can be described as follows:

$$\boldsymbol{h}_l^{(i)} = \overline{\boldsymbol{A}}_l \boldsymbol{h}_{l-1}^{(i)} + \overline{\boldsymbol{B}}_l u_l^{(i)} \in \mathbb{R}^{d_h \times 1} \quad (1a) \qquad \overline{\boldsymbol{A}}_l = \exp(\Delta_l \boldsymbol{A}) \in \mathbb{R}^{d_h \times d_h} \quad (2a)$$

$$o_l^{(i)} = \boldsymbol{C}_l^\top \boldsymbol{h}_l^{(i)}, \quad \boldsymbol{C}_l \in \mathbb{R}^{d_h \times 1} \quad (1b) \qquad \overline{\boldsymbol{B}}_l = (\Delta_l \boldsymbol{A})^{-1}(\exp(\Delta_l \boldsymbol{A}) - \boldsymbol{I})\Delta_l \boldsymbol{B}_l \in \mathbb{R}^{d_h \times 1} \quad (2b)$$

$$\boldsymbol{B}_l = \boldsymbol{W}_B \boldsymbol{u}_l + \boldsymbol{b}_B \quad (3) \qquad \boldsymbol{C}_l = \boldsymbol{W}_C \boldsymbol{u}_l + \boldsymbol{b}_C \quad (4) \qquad \Delta_l = \text{softplus}(\boldsymbol{w}_\Delta^\top \boldsymbol{u}_l + b_\Delta) \quad (5)$$

$$\begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_N & \boldsymbol{x}_q \\ y_1 & y_2 & \cdots & y_N & \boxed{0} \end{bmatrix} \xrightarrow{\text{Mamba}} \begin{bmatrix} \boldsymbol{o}_1^{(1:d)} & \cdots & \boldsymbol{o}_{N+1}^{(1:d)} \\ \boldsymbol{o}_1^{(d+1)} & \cdots & \boxed{\boldsymbol{o}_{N+1}^{(d+1)}} \end{bmatrix}$$

input output

Use $\hat{y}_q = \boldsymbol{o}_{N+1}^{(d+1)}$ as the prediction for y_q

Assumptions

Assumption 4.1 *(1) Matrix $\mathbf{A} = -\mathbf{I}_{d_h}$. (2) The vector $\mathbf{w}_\Delta$ is fixed as zero $\mathbf{0}$, and b_Δ is fixed as* $\ln(\exp((\ln 2)/N) - 1)$. *(3) Matrices $\mathbf{W}_B$, $\mathbf{W}_C$ are initialized with entries drawn i.i.d. from the standard Gaussian distribution $\mathcal{N}(0,1)$. (4) The hidden state dimension satisfies: $d_h = \widetilde{\Omega}(d^2)$. (5) The learning rate satisfies: $\eta = O(d^{-2}d_h^{-1})$. (6) Bias vectors $\mathbf{b}_B$, $\mathbf{b}_C$ are initialized as zero $\mathbf{0}$. (7) Token length $N = \Omega(d)$.*

B，C are initialized with Gaussian distribution

Training Algorithm

Mse Loss + Gradient Descent

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}_{1:N}, \boldsymbol{x}_q, \boldsymbol{w}}\left[\frac{1}{2}(\hat{y}_q - y_q)^2\right]$$

$$\boldsymbol{\theta}'(t+1) = \boldsymbol{\theta}'(t) - \eta \nabla_{\boldsymbol{\theta}'} \mathcal{L}(\boldsymbol{\theta}(t))$$

Main Results

Theorem 4.1 *Under Assumption 4.1, if the Mamba is trained with gradient descent, and given a new prompt $(\boldsymbol{e}_1, \ldots, \boldsymbol{e}_N, \boldsymbol{e}_q)$, then with probability at least $1-\delta$ for some $\delta \in (0,1)$, the trainable parameters $\boldsymbol{\theta}'(t) = \{\boldsymbol{W}_B(t), \boldsymbol{W}_C(t), \boldsymbol{b}_B(t), \boldsymbol{b}_C(t)\}$ converge as $t \to \infty$ to parameters that satisfies:*

(a) Projected hidden state: $(\boldsymbol{W}_C^\top)_{[1:d,:]}(t)\boldsymbol{h}_l^{(d+1)} = \alpha(\boldsymbol{W}_C^\top(t))_{[1:d,:]}\boldsymbol{h}_{l-1}^{(d+1)} + (1-\alpha)\beta y_l \boldsymbol{x}_l$,

(b) Prediction for target: $\hat{y}_q = \boldsymbol{x}_q^\top \sum_{i=0}^{N-1}(1-\alpha)\alpha^{i+1}\beta y_{N-i}\boldsymbol{x}_{N-i}$,

(c) Population loss: $\mathcal{L}(\boldsymbol{\theta}(t)) \le \frac{3d(d+1)}{2N}$,

where $\alpha = \exp((-\ln 2)/N)$, $\beta = \frac{2(1+\alpha)}{\alpha\left(3(1-\alpha)d + 4 - 2\alpha\right)}$.

Theorem 4.1 (a)

Mamba emulates online GD for ICL

Theorem 4.1 (b)

The prediction for y_q is a linear combination of $y_l \boldsymbol{x}_q^T \boldsymbol{x}_l$

Theorem 4.1 (c)

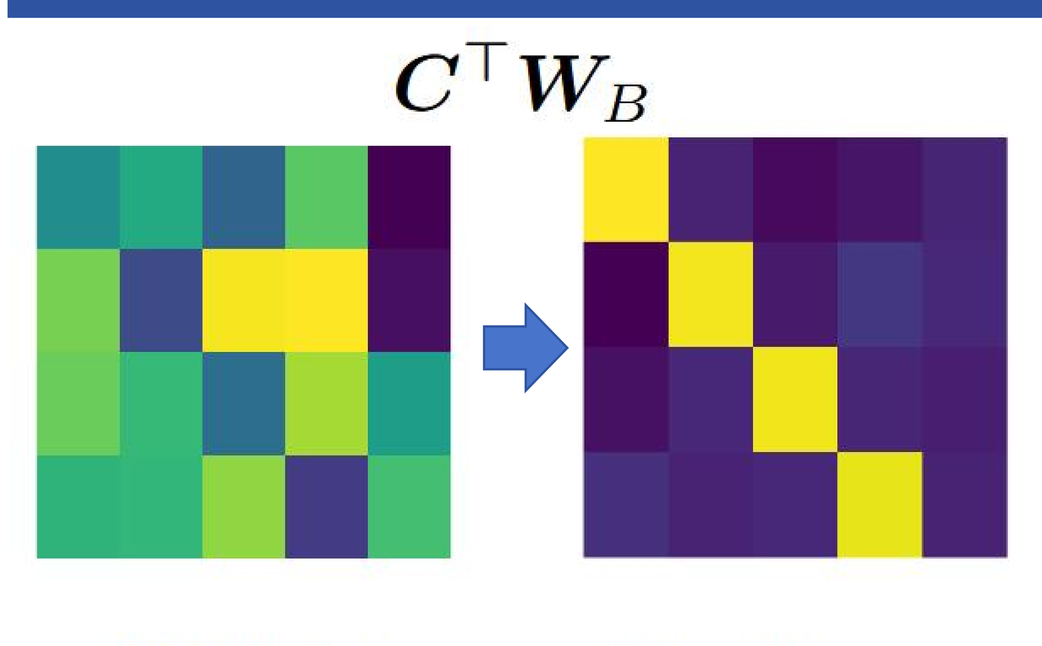
The Loss bound is linearly related to the context length *N*

Compare with Transformer

$$\textbf{Transformer}(\boldsymbol{e}_1, \ldots, \boldsymbol{e}_N, \boldsymbol{e}_q) \approx \boldsymbol{x}_q^\top \left(\frac{1}{N}\sum_{i=1}^N y_i \boldsymbol{x}_i\right) \approx \boldsymbol{x}_q^\top \boldsymbol{w}.$$

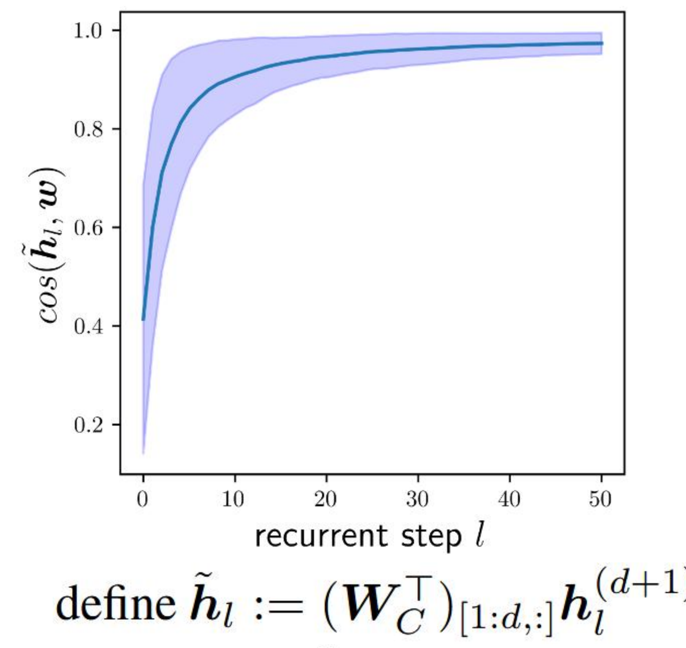
Transformer simulates a single step of GD

Experimental Results



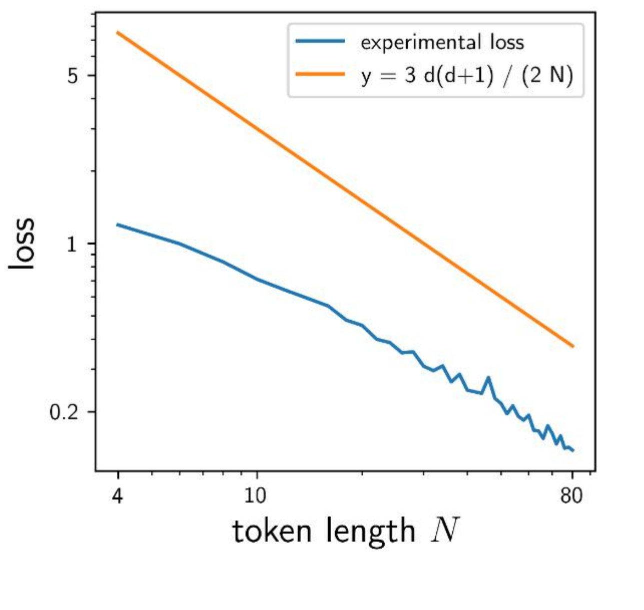
(a) Initialization (b) Trained parameter

A Gaussian initialized Mamba can converge to a solution that can perform ICL



define $\tilde{\boldsymbol{h}}_l := (\boldsymbol{W}_C^\top)_{[1:d,:]}\boldsymbol{h}_l^{(d+1)}$

The direction of $\tilde{\boldsymbol{h}}_l$ converges toward $\boldsymbol{w}$ as mamba processes multiple prompts



The loss has a linear upper bound, which is comparable to Transformer

Insights & takeaways & future work

- The idea of "**online GD**" can provide insights into designing new architectures
- The theoretical analysis can be extended to **multi-layer Mamba** containing nonlinear layers
- The theoretical analysis can be extended to more **complex and diverse ICL tasks** beyond linear regression

Contact

paper LinkedIn Wechat
email: jiaruij@outlook.com