Learning Theory of Transformers: Generalization and Optimization of In-Context Learning

Taiji Suzuki

The University of Tokyo / AIP-RIKEN

(Deep learning theory team)

4th/Dec/2024

Simons Institute Workshop

Transformer architecture

- Transformer

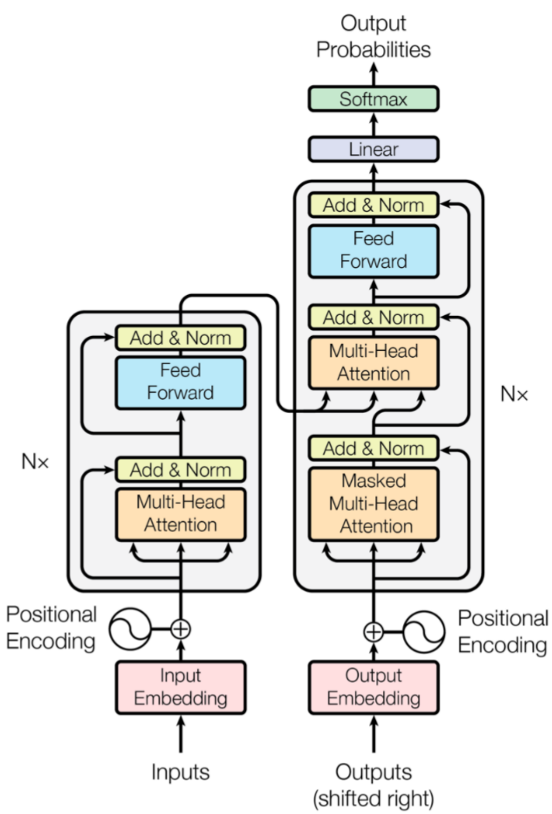
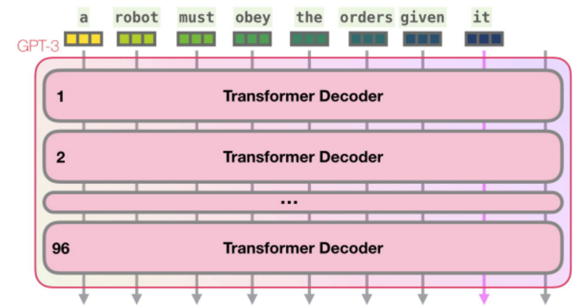


Figure 1: The Transformer - model architecture.

[Vaswani et al.: Attention is All you Need. NIPS2017]

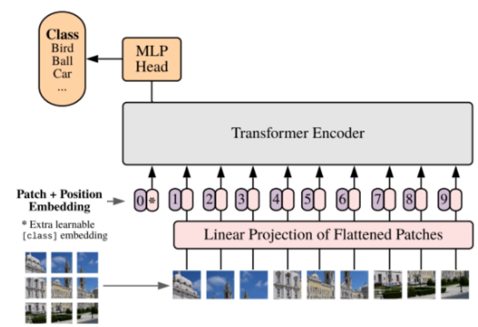
LLM (e.g., GPT3/GPT4)



[Alammar: How GPT3 Works - Visualizations and Animations, https://jalammar.github.io/how-gpt3-works-visualizations-animations/]

[Brown et al. “Language Models are Few-Shot Learners”, NeurIPS2020]

Vision tasks (e.g., ViT)



[Dosovitskiy et al.: An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. arXiv:2010.11929. ICLR2021]

LLM

Please explain what is ChatGPT

ChatGPT is an artificial intelligence (AI) language model created by OpenAI. It is designed to understand and generate human-like text based on the input it receives. Here's a detailed breakdown of what ChatGPT is and what it does:

1. Purpose

- ChatGPT is primarily used for conversation and can respond to questions, provide explanations, write creatively, assist with tasks, and more.
- It serves as a tool for various applications, such as customer support, learning assistance, idea brainstorming, and general information retrieval.

2. How It Works

- **Training:** ChatGPT is trained using a mach learning technique called reinforcement learning from human feedback (RLHF), which refines its ability to respond in helpful, accurate, and safe

[OpenAI, ChatGPT 4o]

Why does it work so well?

In-context learning

Pretrained Large Language Models (LLMs) have significant ability of In-Context Learning (ICL) [Brown et al., 2020].

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

context

1,1 -> 2
2,3 -> 5
8,13 -> 21
6,0 -> 6
10,1 -> 11
5,27 -> ?

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.

Question

ChatGPT

In-context learning

Pretrained Large Language Models (LLMs) have significant ability of In-Context Learning (ICL) [Brown et al., 2020].

Question

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

context

left -> right
dark -> light
short -> long
small -> big
up -> ?

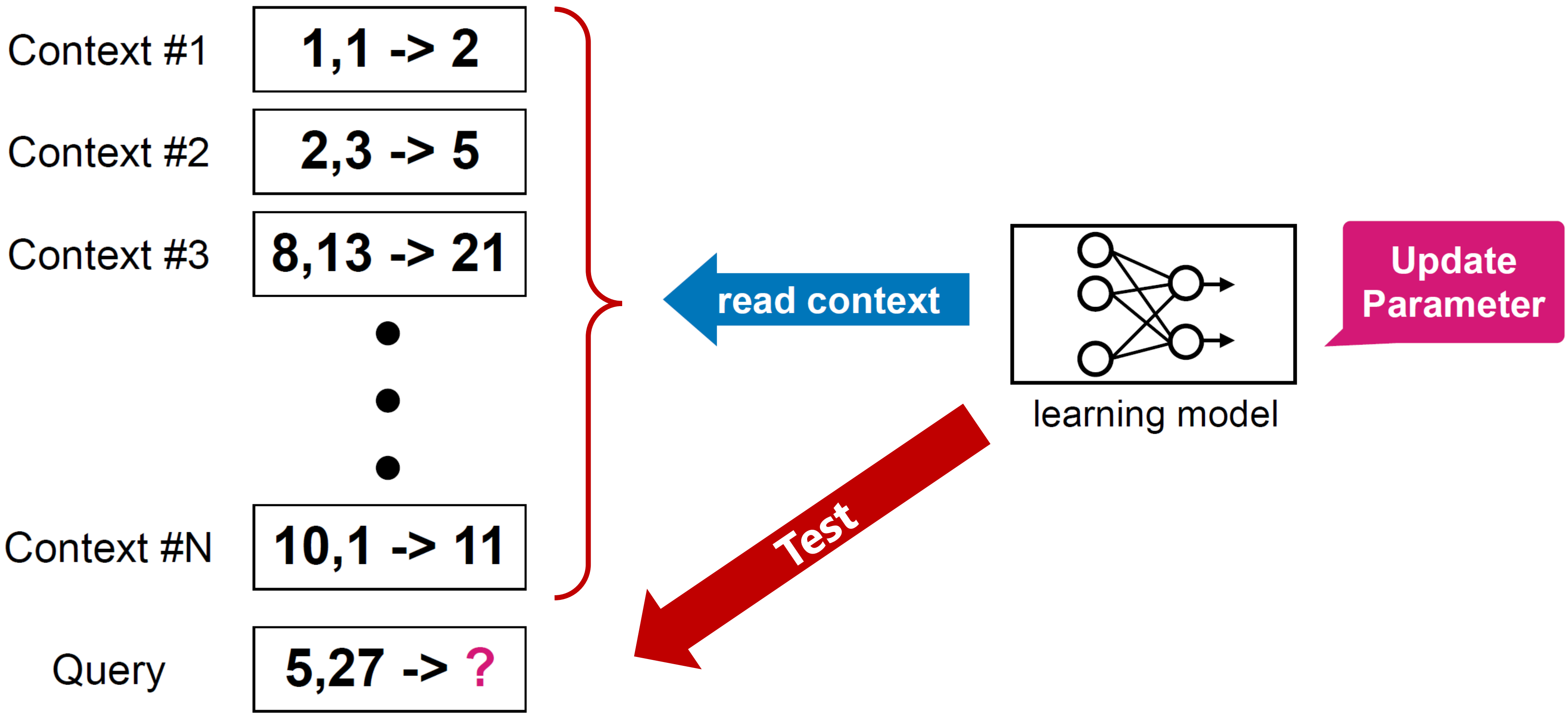
ChatGPT

The pattern in the given pairs of words seems to be antonyms:

So, the word that fits in the '?' is "down".

Fine tuning method

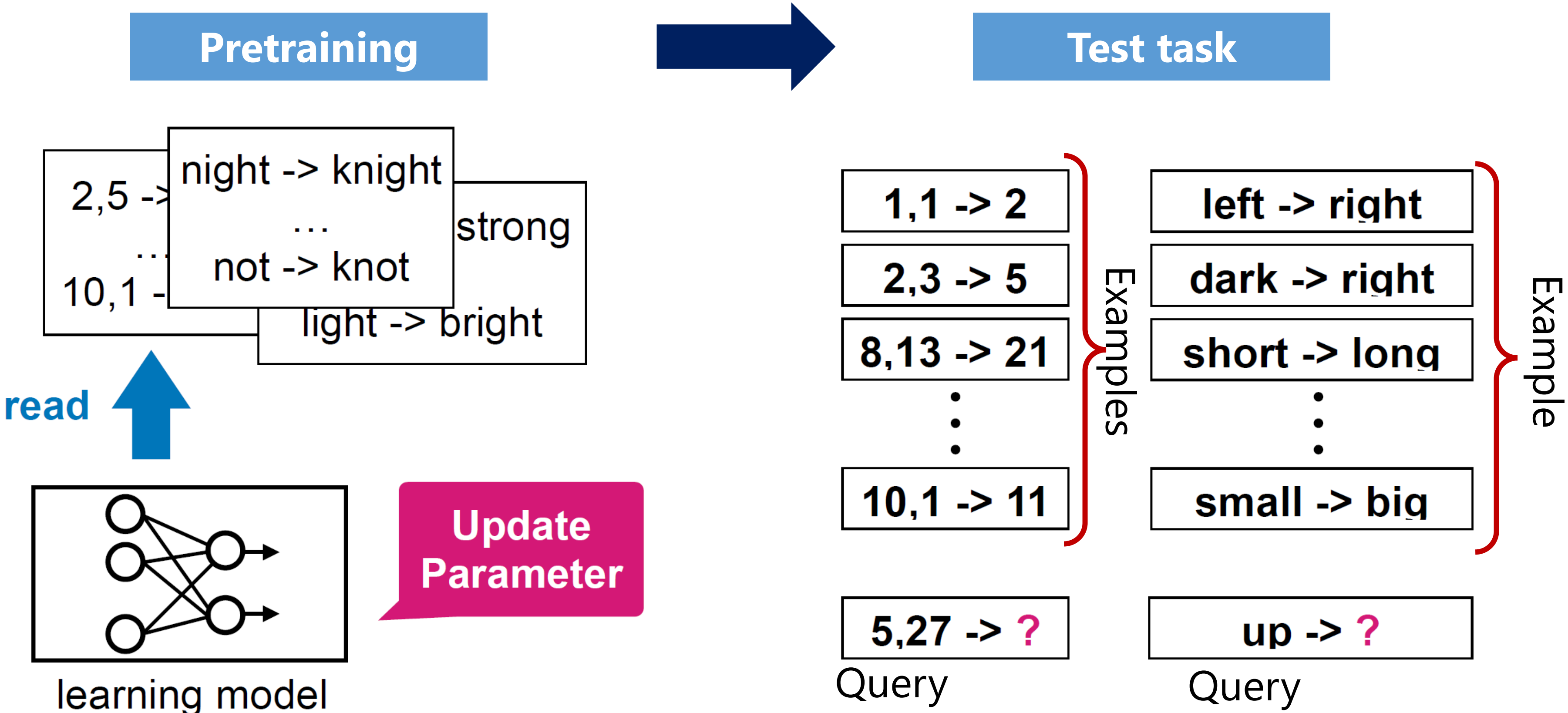
Traditional “fine tuning” approach



(e.g., RLHF)

In-Context learning

ICL is performed **without updating model parameters** unlike the traditional "fine-tuning" regime in the test task.
→ Meta-learning



During pretraining, several tasks are observed to train the model.
→ Task generalization.

Question:
What mechanism allows a Transformer to perform ICL?

Presentation overview

Statistics

Minimax optimality
- Nonparametric analysis
- Approximation error analysis

Optimization

Global optimality of nonlinear feature learning
- Mean field limit
- Strict saddle

Statistics/Optimization

Feature learning with one step GD
- Single index model
- Information exponent
- Advantage of pre-training

- [Minimax optimality and approximation error bound] Kim, Nakamaki, Suzuki: Transformers are Minimax Optimal Nonparametric In-Context Learners. NeurIPS2024
- [Optimization in mean field limit] Kim, Suzuki: Transformers Learn Nonlinear Features In Context: Nonconvex Mean-field Dynamics on the Attention Landscape. ICML2024 (arXiv:2402.01258).
- [Identifying low dimensional subspace with information exponent k] Oko, Song, Suzuki, Wu: Transformer efficiently learns low-dimensional functions in context. NeurIPS2024.

Approximation theory/ Statistical analysis

Nonparametric analysis of in-context learning

[Kim, Nakamaki, Suzuki: Transformers are Minimax Optimal Nonparametric In-Context Learners. NeurIPS2024]

Juno Kim

Mathematical formulation of in-context learning

Model: $y_{i,t} = F_t^{\circ}(x_{i,t}) + \epsilon_{i,t} \qquad (i = 1, \ldots, n)$

$t = 1, \ldots, T$: Task index

- The true functions F_t° are different across different tasks.
- F_t° is generated randomly for each task.

Pretraining (T tasks):

$X_t = [x_{1,t}; \ldots; x_{n,t}]$ $\quad x_{\mathrm{qr},t}$

...

$Y_t = [y_{1,t}; \ldots; y_{n,t}]$ $\quad y_{\mathrm{qr},t}$

$\times T$

- ➢ We observe pretraining task data T times.
- ➢ Each task has n data.

Test task (In-context learning):

$X_{T+1} = [x_{1,T+1}; \ldots; x_{n,T+1}]$

...

$Y_{T+1} = [y_{1,T+1}; \ldots; y_{n,T+1}]$

(Implicit) Bayes estimation

- ➢ Learn prior at pretraining
- ➢ Perform posterior inference at the test task

Linear combination of features

Suppose that the true function admits a basis function decomposition:

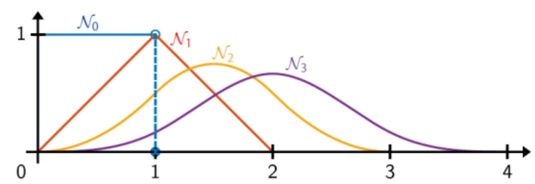
$$F_t^\circ(x) = \beta_t^\top f^\circ(x)$$

where $\beta_t \sim (0, \Sigma)$ and $f^\circ(x) \in \mathbb{R}^\infty$.

- B-Spline (Besov)

$$\mathcal{N}(x) = \begin{cases} 1 & (x \in [0,1]), \\ 0 & (\text{otherwise}) \end{cases}$$

$$\mathcal{N}_m(x) = (\underbrace{\mathcal{N} * \mathcal{N} * \cdots * \mathcal{N}}_{m+1 \text{ times}})(x)$$



Tensor product B-spline:

$$M_{a,b}^d(x) = \prod_{j=1}^d \mathcal{N}_m(2^{a_j} - b_j)$$

$$f_j^\circ(x) = M_{a^{(j)},b^{(j)}}^d(x) \Longrightarrow F_\beta^\circ = \beta^\top f^\circ \in B_{2,2}^\alpha$$

- Fourier (Sobolev, γ-smooth)

$$f_j^\circ(x) = \prod_{k=1}^\infty \sqrt{2}\cos(2\pi 2^{s_{j,k}} x_k - \delta_{j,k}\pi/2) \Longrightarrow F_\beta^\circ \in \mathcal{F}_{2,2}^\gamma([0,1]^\infty)$$

γ-smooth function class for $d = \infty$ [Okumoto&Suzuki,ICLR2022], [Takakura&Suzuki, ICML2024]

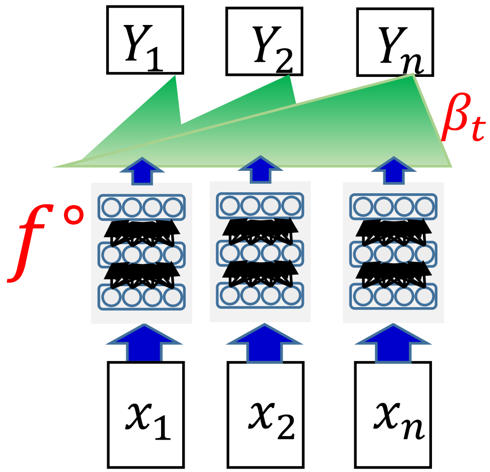
Feature map and linear coeff

$$F_t^\circ(x) = \beta_t^\top f^\circ(x)$$

- **Pretraining: Learning feature map $[f^\circ]$**
  - Fourier basis, B-Spline
  - Independent of context (t)
  - Obtain the most "efficient" basis to represent data
    → Internal layers

- **Good representation**
- **Distribution of β_t**

- **In-context learning: Estimating coefficient $[\beta_t]$**
  - Dependent on context (t)
  - Estimate the context β_t from the instruction (Attention)
    → Attention layer



✓ Guo et al. 2023 and von Oswald et al. 2023 observed that real Transformers extract nonlinear features at lower layers and perform linear regression deeper layers.
→ It is not like performing gradient descent at every layer as in Bai et al. 2023.

Transformer model

A. Nonlinear feature map (FNN)

We approximate the infinite dimensional nonlinear feature map f° by DNN:

$$\phi : \mathbb{R}^d \to \mathbb{R}^N \quad (f^\circ \simeq \phi)$$

Deep neural network (nonlinear feature map)

B-1. Soft-max attention model

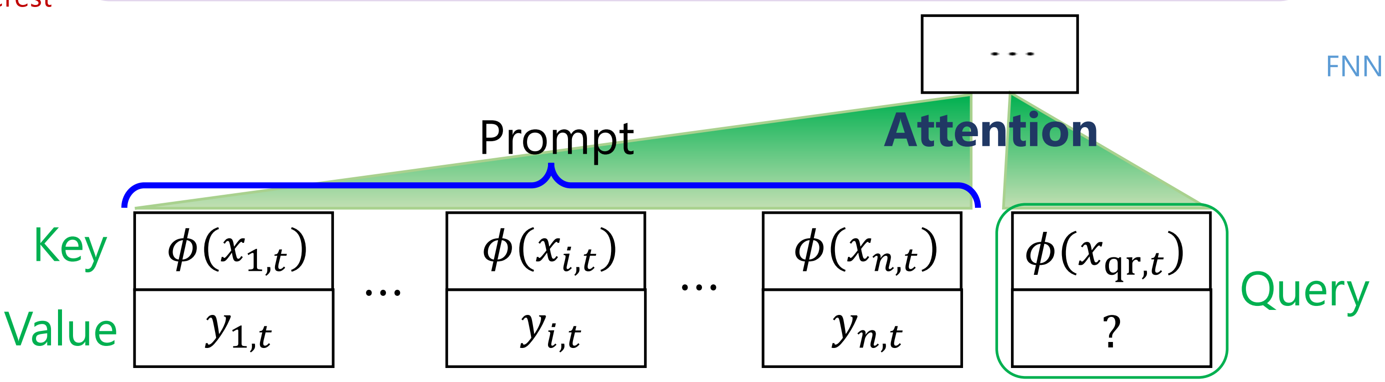
$$\sum_{i=1}^{n} \underbrace{y_{i,t}}_{\text{Value}} \frac{\exp(\underbrace{\phi(x_{i,t})}_{\text{Key}}{}^\top KQ \underbrace{\phi(x_{\mathrm{qr},t})}_{\text{Query}})}{\sum_{i'=1}^{n} \exp(\phi(x_{i',t})^\top KQ\phi(x_{\mathrm{qr},t}))} \xrightarrow{\text{Predict}} y_{\mathrm{qr},t}$$

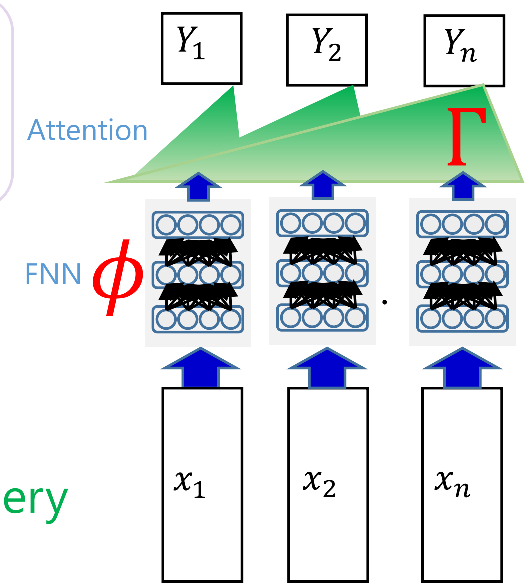
B-2. Linear attention model

[Ahn et al.: Linear attention is (maybe) all you need (to understand transformer optimization). arXiv:2310.01082]

Today's interest

$$\frac{1}{n}\sum_{i=1}^{n} y_{i,t}\phi(x_{i,t})^\top \overset{\Gamma}{\boxed{KQ}}\phi(x_{\mathrm{qr},t})$$





※ In practice, each token should be a couple $(\phi(x), y)$. But, for this theoretical research, we simplify the Q, K, V to a specific form

In-Context Learning (ICL) risk

(Linear) attention can implement linear regression:

$$Y^\top \phi(X)(\phi(X)^\top\phi(X)+n\Lambda)^{-1}\phi(x_{\mathrm{qr}}) = \overbrace{\frac{1}{n}\sum_{i=1}^{n} y_i \phi(x_i)^\top \boxed{\left(\frac{\phi(X)^\top\phi(X)}{n}+\Lambda\right)^{-1}}}^{\beta^\top}\phi(x_{\mathrm{qr}})$$

$\simeq \Gamma$ (prior information)

Carefully chosen Γ yields (nearly) Bayes optimal estimator.

[Gang et al. 2022; Akyurek et al. 2023; Zhang et al. 2023; Ahn et al. 2023; Mahankali et al., 2023; Wu et al. 2024]

Empirical ICL risk :

$$\widehat{\mathcal{L}}(\phi,\Gamma) := \frac{1}{T}\sum_{t=1}^{T}\left(y_{\mathrm{qr},t} - \frac{1}{n}\sum_{i=1}^{n} y_{i,t}\phi(x_{i,t})^\top\Gamma\phi(x_{\mathrm{qr},t})\right)^2$$

→ **Minimize with respect to ϕ (feature map) and Γ (attention param).**

The expected ICL risk:

$\mathcal{L}(\phi,\Gamma)$: … $_{qr}))^2]$

(where F …

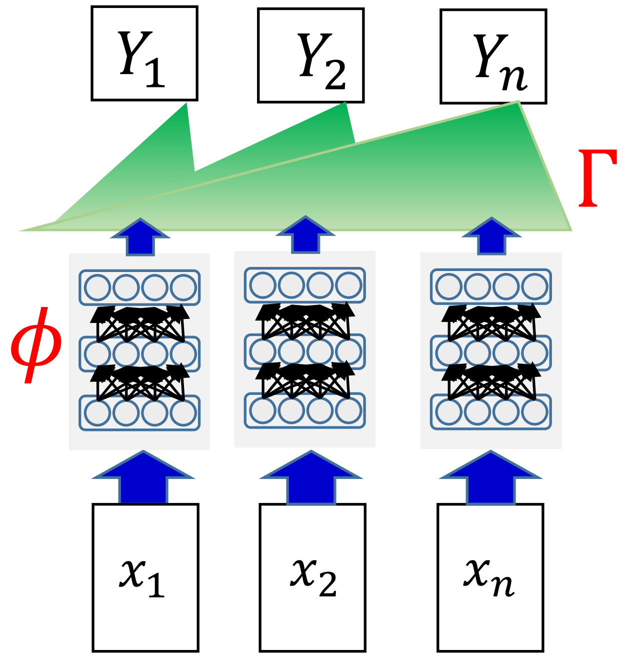
Question :
- Can we obtain "optimal" expected risk?
- What is the benefit of ICL?

Empirical risk minimizer

Empirical risk minimizer:

$$\min_{\Gamma \in \mathbb{R}^{N \times N}, \phi \in \mathrm{DNN}} \widehat{\mathcal{L}}(\phi, \Gamma) := \frac{1}{T} \sum_{t=1}^{T} \left(y_{\mathrm{qr},t} - \frac{1}{n} \sum_{i=1}^{n} y_{i,t} \phi(x_{i,t})^\top \Gamma \phi(x_{\mathrm{qr},t}) \right)^2$$

$$\mathcal{F}_N := \{\phi : \mathbb{R}^d \to \mathbb{R}^N \mid \phi \in \mathrm{DNN}\}$$



Predictive error bound

Empirical risk minimizer:

$$(\hat{\phi}, \hat{\Gamma}) \leftarrow \underset{\Gamma \in \mathbb{R}^{N\times N}, \phi \in \mathrm{DNN}}{\arg\min} \widehat{\mathcal{L}}(\phi, \Gamma) := \frac{1}{T}\sum_{t=1}^{T}\left(y_{\mathrm{qr},t} - \frac{1}{n}\sum_{i=1}^{n} y_{i,t}\phi(x_{i,t})^\top \Gamma \phi(x_{\mathrm{qr},t})\right)^2$$

$$\mathcal{F}_N := \{\phi : \mathbb{R}^d \to \mathbb{R}^N \mid \phi \in \mathrm{DNN\ with\ presprcified\ hyper\text{-}param}\}$$

Assumption (informal)

1. $\mathbb{E}[\beta_{t,j}^2] \lesssim j^{-2s-1-\epsilon}$ (Complexity of function space)
2. $\inf_{\phi \in \mathcal{F}_N} \max_{1 \le j \le N} \|f_j^\circ - \phi_j\|_\infty \lesssim \delta_N$ (Approx. error of each basis)
3. $\|\sum_{j=1}^{k} (f_j^\circ)^2\|_\infty \lesssim k^{2r}$ (Bases are bounded)
4. $(f_j^\circ)_{j=1}^{\infty}$ are "near" orthonormal (Bases are almost orthogonal to each other)

Thm. (ICL risk bound; Kim, Nakamaki, TS, NeurIPS2024)

$$\mathbb{E}[\mathcal{L}(\hat{\phi}, \hat{\Gamma})] \lesssim N^{-2s} + N^2\delta_N^4 + N^{2r+1}\delta_N^2$$

Feature approximation error

$$+ \frac{N}{n} + \frac{N^{2r}}{n}\log(N) + \frac{N^{4r}}{n^2}\log^2(N)$$

In-context generalization gap

$$+ \frac{1}{T}\left(N^2\log(\epsilon^{-1}) + \log(\mathcal{N}(\tfrac{\epsilon}{\sqrt{N}}, \mathcal{F}_N, \|\cdot\|_\infty))\right) + \epsilon$$

Covering number of DNN

Pretraining generalization to estimate basis functions

Examples

- **Example (B-spline basis; f_j° is B-spline→Besov/Sobolev space):**

Estimator 1: $$\mathbb{E}[\mathcal{L}(\hat{\phi},\hat{\Gamma})] \lesssim N^{-2s} + \frac{N\log(N)}{n} + \frac{N^2\log(N)}{T}$$

Bias-variance trade-off

$$\Rightarrow \boxed{\mathbb{E}[\mathcal{L}(\hat{\phi},\hat{\Gamma})] \lesssim n^{-\frac{2s}{2s+1}} + \frac{n^{\frac{2}{2s+1}}}{T} \wedge T^{-\frac{2s}{2(s+1)}} + \frac{T^{\frac{2}{2(s+1)}}}{n}}$$

Minimax optimal w.r.t. n (if T is large)

Small T: memorization
Large T: generalization

- **Example (Holder class basis; $f_j^\circ \in H^{\alpha'}(\mathbb{R}^d)$):**

Estimator 2 (Γ is restricted to a diagonal matrix):

$$\mathbb{E}[\mathcal{L}(\hat{\phi},\hat{\Gamma})] \lesssim N^{-2s} + \frac{N\log(N)}{n} + \frac{N^{1+\frac{d}{\alpha'}(1+s)}\log(N)}{T}$$

$$\Rightarrow \boxed{\mathbb{E}[\mathcal{L}(\hat{\phi},\hat{\Gamma})] \lesssim n^{-\frac{2s}{2s+1}} + n^{\frac{1+\frac{d}{\alpha'}(1+s)}{2s+1}} T^{-1}}$$

If there is no-pretraining, the minimax lower bound is

$$\boxed{\mathbb{E}[\mathcal{L}(\hat{\phi},\hat{\Gamma})] \gtrsim \max\{n^{-\frac{2s}{2s+1}}, n^{-\frac{2\alpha'}{2\alpha'+d}}\}}$$

With many pretraining data, the pretrained model can outperform direct estimator.

Pretraining improves the error by estimating the bases in the pretraining phase

Mini-max lower bound

$$\mathcal{L}(\hat{f}) := \mathbb{E}_{\beta, x_{\mathrm{qr}}}\left[\left(F^\circ_\beta(x_{\mathrm{qr}}) - \hat{f}(x_{\mathrm{qr}})\right)^2\right]$$

$\hat{f}$: depending on the pretraining data $(x_{t,i}, y_{t,i})_{t=1,i=1}^{T,n}$ and new task data $(x_{T+1,i}, y_{T+1,i})_{i=1}^{n}$.

Minimax risk: $\inf_{\hat{f}} \sup_{f^\circ \in \mathcal{F}^\circ} \mathbb{E}[\mathcal{L}(\hat{f})]$

Information theoretic lower bound:

Prop. (ICL risk lower bound)

$$\inf_{\hat{f}} \sup_{f^\circ \in \mathcal{F}^\circ} \mathbb{E}[\mathcal{L}(\hat{f})] \gtrsim \delta_{n,T}^2 \geq \epsilon_{1,n}^2 + \epsilon_{2,n}^2$$

where

(log-covering number) Complexity to estimate

$$\begin{cases} \epsilon_{1,n}^2 \simeq \dfrac{V(\epsilon_{1,n}, \mathcal{F}^\circ)}{nT} & \text{: the basis function } f^\circ \\ \epsilon_{2,n}^2 \simeq \dfrac{\epsilon_{2,n}^{-1/s}}{n} & \text{: coefficient } \beta_T \end{cases}$$

We consider f° as a random variable "uniformly" distributed on a model:

$$\inf_{\hat{f}} \sup_{f^\circ \in \mathcal{F}^\circ} \mathbb{E}[\mathcal{L}(\hat{f})] \gtrsim \delta^2 \left(1 - \frac{I(D_{1:T+1} \| (f^\circ, \beta_{T+1})) + \log(2)}{\log(\mathcal{N}(\delta, \{F^\circ_\beta\}))}\right)$$

Concrete example

Optimal rate when the basis is known.

Complexity to estimate the basis

(log-covering number)

$$\inf_{\hat{f}} \sup_{f^\circ \in \mathcal{F}^\circ} \mathbb{E}[\mathcal{L}(\hat{f})] \gtrsim n^{-\frac{2s}{2s+1}} + \frac{V(\epsilon_{1,n}, \mathcal{F}^\circ)}{nT}$$

where $\epsilon_{1,n}^2 \simeq \frac{V(\epsilon_{1,n}, \mathcal{F}^\circ)}{nT}$

- **Basis functions in Holder space** ($f_j^\circ \in H^{\alpha'}(\mathbb{R}^d)$): $\frac{V(\epsilon_{1,n}, \mathcal{F}^\circ)}{nT} \simeq \frac{\epsilon_{1,n}^{-d/\alpha'}}{nT}$

$$\boxed{\inf_{\hat{f}} \sup_{f^\circ \in \mathcal{F}^\circ} \mathbb{E}[\mathcal{L}(\hat{f})] \gtrsim n^{-\frac{2s}{2s+1}} + (nT)^{-\frac{2\alpha'}{2\alpha'+d}}}$$

Suppose that $\alpha'/d < s$, then

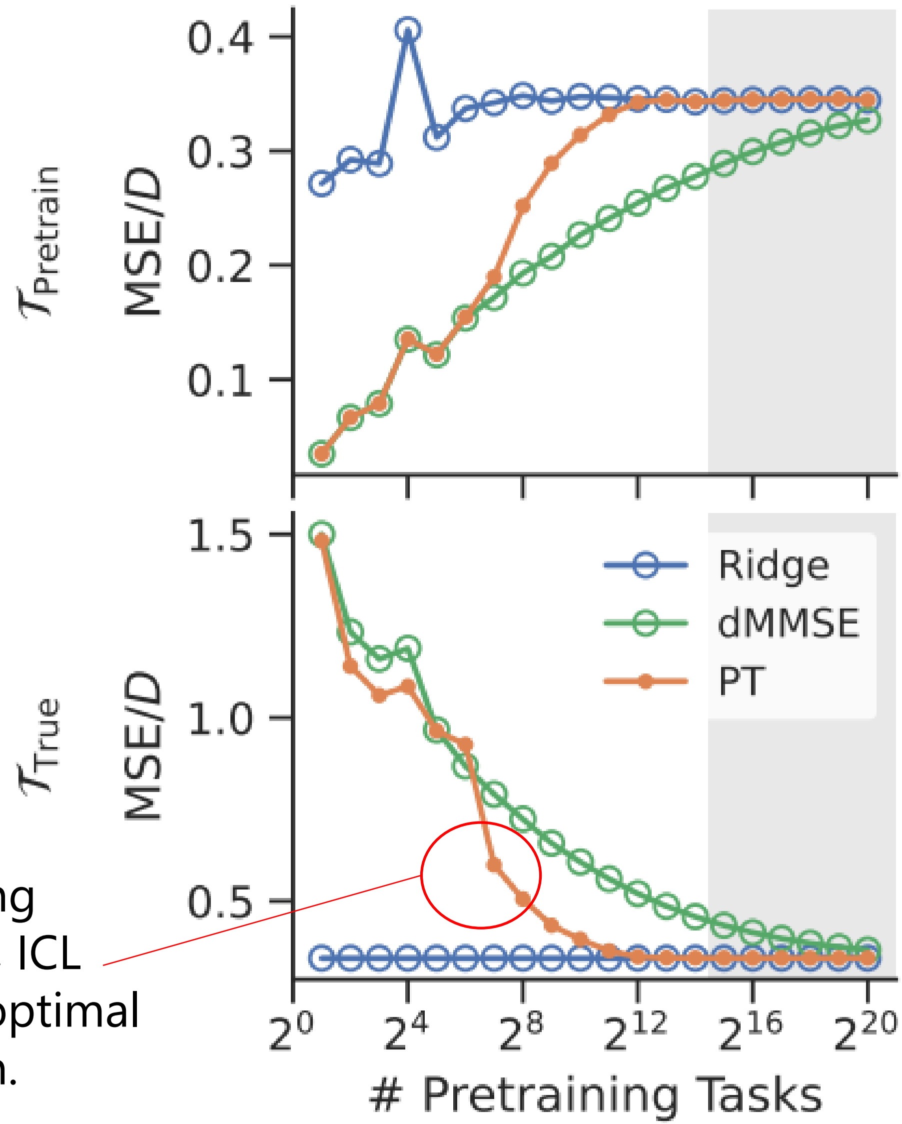
No pretraining ($T = 1$): $n^{-\frac{2\alpha'}{2\alpha'+d}}$

$\gg$

Pretraining setting ($T \gg 1$): $n^{-\frac{2s}{2s+1}}$

When T is large, pretraining can give better generalization for test instruction than learning from scratch

Task diversity matters



If # of pretraining tasks is enough, ICL coincides with optimal ridge regression.

[Raventós, Paul, Chen, Ganguli: Pretraining task diversity and the emergence of non-Bayesian in-context learning for regression. 2023]

Presentation overview

Statistics

Minimax optimality
- Nonparametric analysis
- Approximation error analysis

Optimization

Global optimality of nonlinear feature learning
- Mean field limit
- Strict saddle

Statistics/Optimization

Feature learning with one step GD
- Single index model
- Information exponent
- Advantage of pre-training

- [Minimax optimality and approximation error bound] Kim, Nakamaki, Suzuki: Transformers are Minimax Optimal Nonparametric In-Context Learners. NeurIPS2024
- [Optimization in mean field limit] Kim, Suzuki: Transformers Learn Nonlinear Features In Context: Nonconvex Mean-field Dynamics on the Attention Landscape. ICML2024 (arXiv:2402.01258).
- [Identifying low dimensional subspace with information exponent k] Oko, Song, Suzuki, Wu: Transformer efficiently learns low-dimensional functions in context. NeurIPS2024.

So far, we have considered approximation theory.
From now on, we discuss optimization theory.

Global optimality of GD for in-context learning

[Kim, Suzuki: Transformers Learn Nonlinear Features In Context: Nonconvex Mean-field Dynamics on the Attention Landscape. **ICML2024, oral presentation** (arXiv:2402.01258)]

Juno Kim

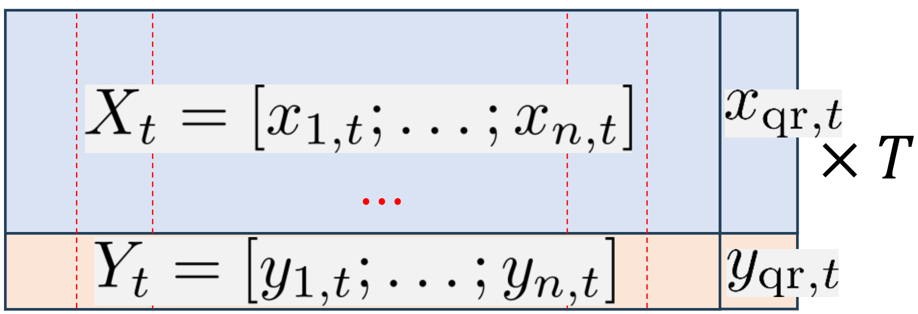
Mathematical formulation of in-context learning

Model: $y_{i,t} = F_t^{\circ}(x_{i,t}) + \epsilon_{i,t} \qquad (i = 1, \ldots, n)$

$t = 1, \ldots, T$: Task index

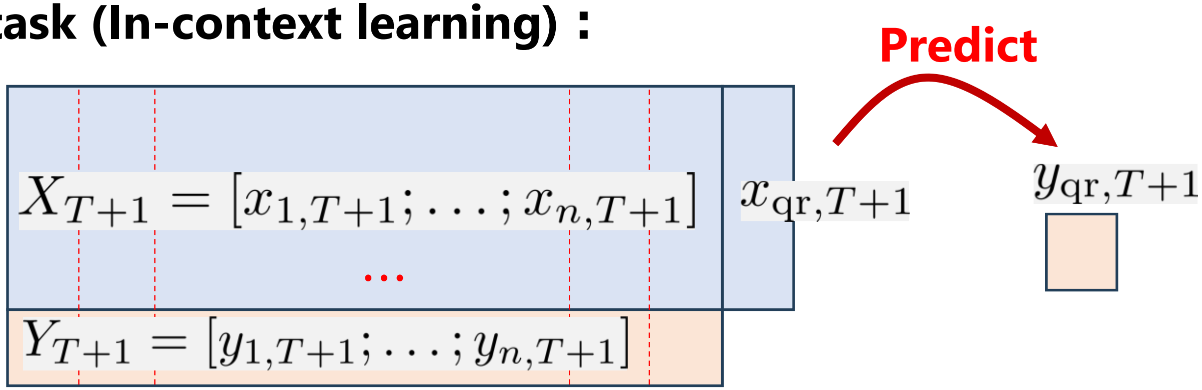
- The true functions F_t are different across different tasks.
- F_t° is generated randomly for each task.

Pretraining (T tasks):



- ➢ We observe pretraining task data T times.
- ➢ Each task has n data.

Test task (In-context learning):



Model: Nonlinear feature

Linear model with nonlinear features:

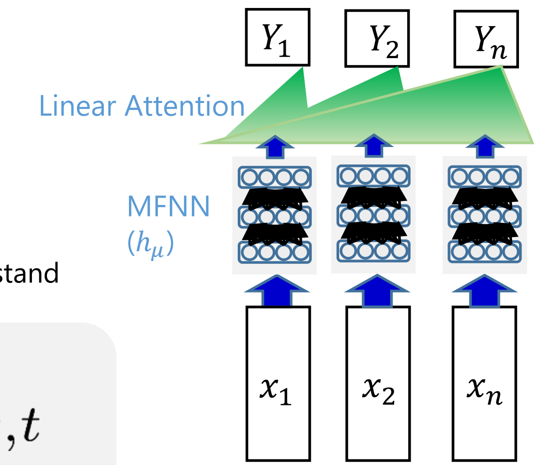
$$F_t^\circ(x) = v_t^\top f^\circ(x) \qquad \text{where } v_t \sim N(0, I) \text{ and } f^\circ(x) \in \mathbb{R}^k.$$

We want to estimate the nonlinear feature f° by pretraining.

- **Mean field neural network (Barron class):**

$$h_\mu(x) = \int h_\theta(x) \mathrm{d}\mu(\theta) \in \mathbb{R}^k$$

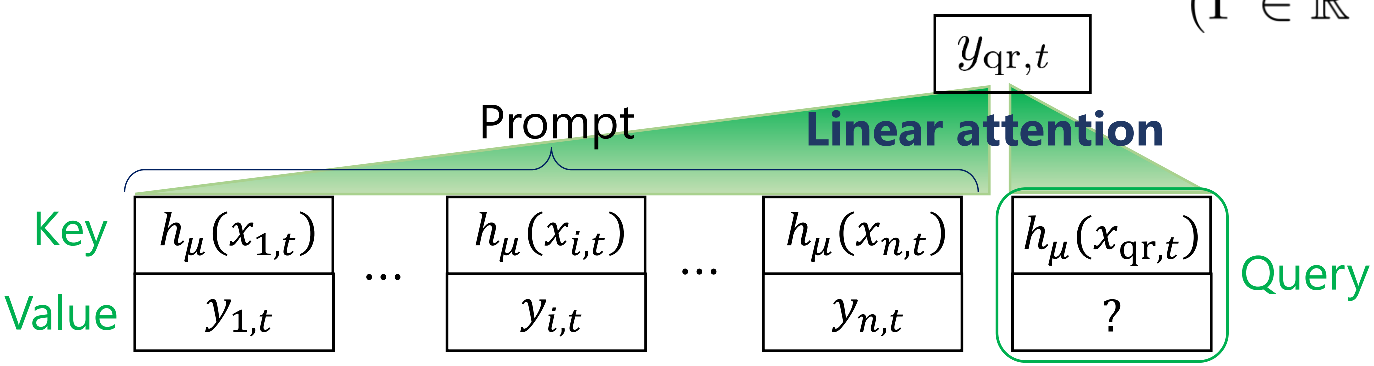
$$h_\theta(x) = \mathbf{a}\sigma(\mathbf{w}^\top x) \quad (\theta = (\mathbf{a}, \mathbf{w}) \in \mathbb{R}^k \times \mathbb{R}^d)$$



- **Linear attention:** [Ahn et al.: Linear attention is (maybe) all you need (to understand transformer optimization). arXiv:2310.01082]

$$\frac{1}{n}\sum_{i=1}^{n} \underbrace{y_{i,t}}_{\text{Value}} \underbrace{h_\mu(x_{i,t})^\top}_{\text{Key}} \Gamma \underbrace{h_\mu(x_{\mathrm{qr},t})}_{\text{Query}} \xrightarrow{\text{Predict}} y_{\mathrm{qr},t}$$

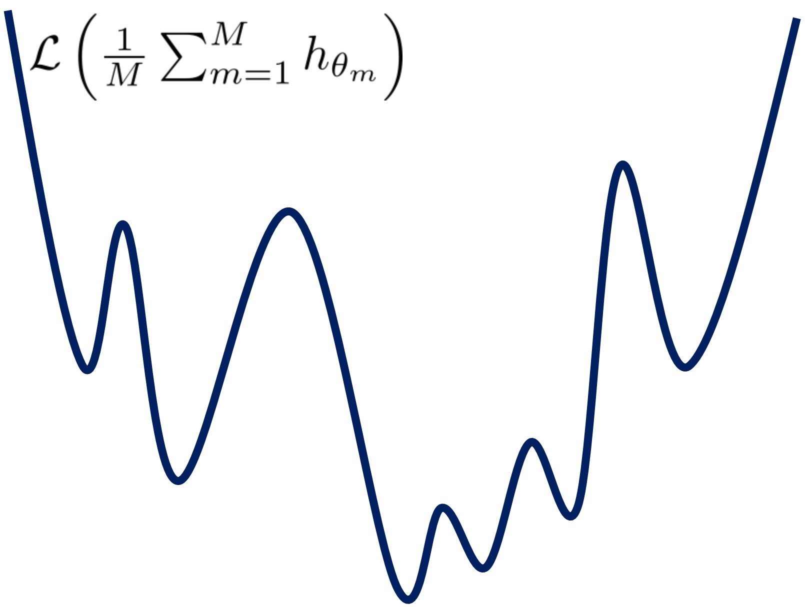
$$(\Gamma \in \mathbb{R}^{k \times k})$$



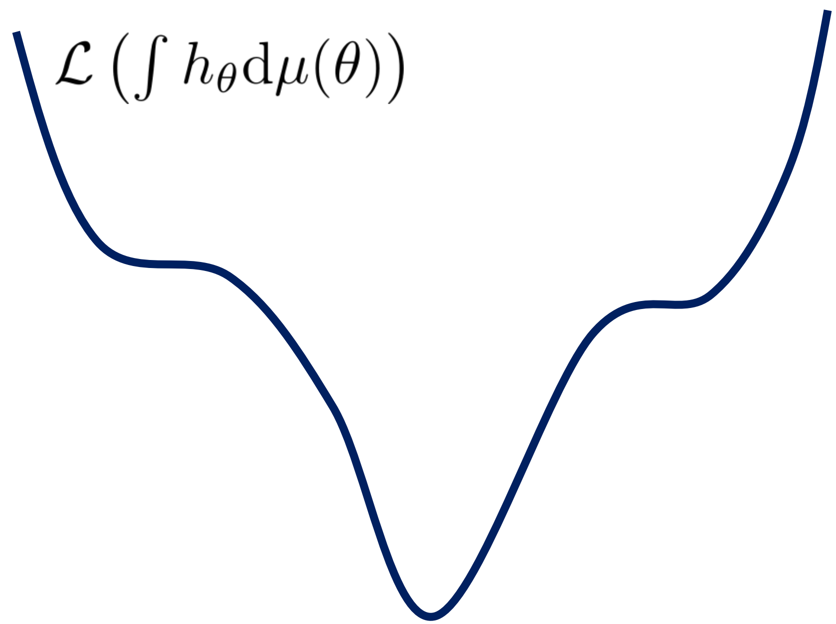
Why mean field?

$$\frac{1}{M}\sum_{m=1}^{M} h_{\theta_m}(x) \to \int h_\theta(x)\mathrm{d}\mu(\theta)$$

(Non-linear w.r.t. $(\theta_m)_{m=1}^M$) (Linear w.r.t. μ)



As a function of θ



As a function of μ

- Mean field Langevin dynamics: [Nitanda,Wu,Suzuki, 2022; Chizat, 2022]
  → Linear convergence with a log-Sobolev inequality for optimizing 2-layer NN.

$$\mathcal{L}(\mu_t) - \mathcal{L}^* \leq \exp(-\lambda\alpha t)(\mathcal{L}(\mu_0) - \mathcal{L}^*)$$

In-Context Learning (ICL) risk

Empirical ICL risk :

$$\widehat{\mathcal{L}}(\mu,\Gamma) := \frac{1}{T}\sum_{t=1}^{T}\left(y_{\mathrm{qr},t} - \frac{1}{n}\sum_{i=1}^{n} y_{i,t} h_\mu(x_{i,t})^\top \Gamma h_\mu(x_{\mathrm{qr},t})\right)^2$$

→ Minimize with respect to μ, Γ.

The expected ICL risk: (Large sample limit: $n \to \infty$ and $T \to \infty$)

$$\mathcal{L}(\mu,\Gamma) := \mathbb{E}_{x_{\mathrm{qr}}}\left[\left\| f^\circ(x_{\mathrm{qr}}) - \mathbb{E}_x[f^\circ(x) h_\mu(x)^\top]\Gamma h_\mu(x_{qr})\right\|^2\right]$$

(note that $y_{i,t} = v_t^\top f^\circ(x_{i,t})$)

Question : Can we optimize μ, Γ by a gradient descent?
(Infinite-dimensional non-convex problem)

There have been many work on optimization guarantee on ICL for **linear model**: Zhang et al., (2023), Mahankali et al. (2023), Guo et al. (2023) to name a few.
Bu, this is a **nonlinear feature learning**.

Two time-scale dynamics

Feature covariance $\Sigma_{\mu,\nu} := \mathbb{E}_X[h_\mu(X)h_\nu^\top(X)]$

$h_\mu(x) := \int h_\theta(x)\mathrm{d}\mu(\theta)$

Assumption (realizability of the true feature)

There exists μ° such that $f^\circ = h_{\mu^\circ}$ and $\Sigma_{\mu^\circ,\mu^\circ} \propto I_k$.

Two time-scale dynamics (Γ is optimized first):

$$\mathcal{L}(\mu) := \min_{\Gamma} \mathcal{L}(\mu, \Gamma) = \min_{\Gamma} \mathbb{E}_{x_{\mathrm{qr}}} \left[\left\| f^\circ(x_{\mathrm{qr}}) - \mathbb{E}_x[f^\circ(x)h_\mu(x)^\top]\Gamma h_\mu(x_{qr}) \right\|^2 \right]$$
$$= \mathbb{E}_{x_{\mathrm{qr}}} \left[\left\| f^\circ(x_{\mathrm{qr}}) - \Sigma_{\mu^\circ,\mu}\Sigma_{\mu,\mu}^{-1} h_\mu(x_{qr}) \right\|^2 \right]$$

- μ is the minimizer iff $h_\mu = R h_{\mu^\circ}$ for an invertible matrix R

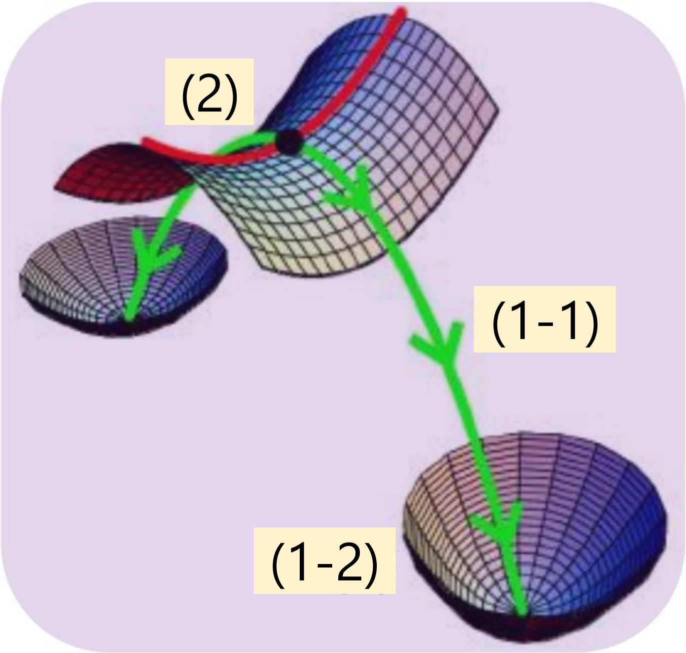
Wasserstein gradient flow to minimize $\mathcal{L}$:

- $\partial_t \mu_t = \nabla \cdot \left(\mu_t \nabla \dfrac{\delta \mathcal{L}(\mu_t)}{\delta \mu} \right)$
- $\dfrac{\mathrm{d}\theta_t}{\mathrm{d}t} = -\nabla \dfrac{\delta \mathcal{L}(\mu_t)}{\delta \mu}(\theta_t) \quad (\mu_t = \mathrm{Law}(\theta_t))$

Strict saddle

- There is no spurious local minima.
- All critical points are saddle and have negative curvature.

Theorem 1 (**Strict saddle** property of the loss landscape)



There exists a **descent direction** or **negative curvature**.

Analogous to matrix completion [Ge et al., 2016, 2017; Bhojanapalli et al. 2016; Li et al., 2019].

Strict saddle

For an orthogonal matrix $\mathbf{R} \in \mathrm{O}(k)$, define $\boldsymbol{R}\#\mu$ as the push-forward of μ along the rotation $\mathbf{R}$: $(a, w) \mapsto (\mathbf{R}a, w)$, i.e., $h_{\mathbf{R}\#\mu} = \mathbf{R}h_\mu$.

Theorem 1 (Strict saddle property of the loss landscape)

If $\mu \in \mathcal{P}$ is not the global minimum, then one of the followings holds:

(1) (1-1) There exists $\mathbf{R} \in \mathrm{conv}(\mathrm{O}(k))$ such that

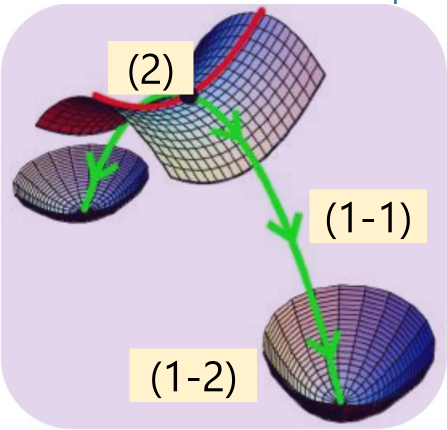
$$\frac{\mathrm{d}}{\mathrm{d}s}\mathcal{L}(\bar{\mu}_s)\Big|_{s=0} < 0 \quad \text{where } \bar{\mu}_s = (1-s)\mu + s\mathbf{R}\sharp\mu^\circ.$$

(1-2) Furthermore, if $0 < \mathcal{L}(\mu) < r^\circ/2$, then

$$\frac{\mathrm{d}}{\mathrm{d}s}\mathcal{L}(\bar{\mu}_s)\Big|_{s=0} \leq -\frac{4}{\|\sigma\|_\infty^2}\mathcal{L}(\mu)\left(\frac{r_0}{2} - \mathcal{L}(\mu)\right)$$

(2) Otherwise,

$$\mathcal{L}(\mu) > \frac{r_0}{2} \quad \text{and} \quad \frac{\mathrm{d}^2\mathcal{L}(\bar{\mu}_s)}{\mathrm{d}s^2}\Big|_{s=0} \leq -\frac{4}{k\|\sigma\|_\infty^2}\mathcal{L}(\mu)^2.$$



There exists a **descent direction** or **negative curvature**.
Analogous to matrix completion [Ge et al., 2016, 2017; Bhojanapal 2019].

Behavior around the critical point

Let the "Hessian" at μ be

$$H_\mu(\theta, \theta') := \nabla_\theta \nabla_{\theta'} \frac{\delta^2 \mathcal{L}(\mu)}{\delta \mu^2}(\theta, \theta')$$

Lemma

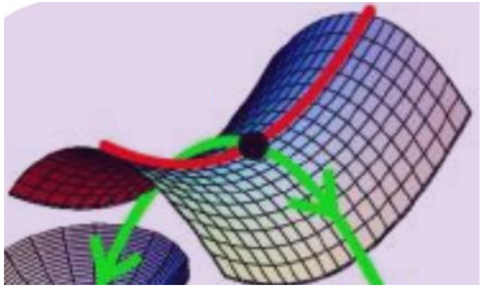
The Wasserstein GF μ_t around a critical point μ^+ can be written as $(\mathrm{id} + \epsilon v_t)\#\mu^+$ where the velocity field v_t follows

$$\partial_t v_t(\theta) = -\int H_{\mu^+}(\theta, \theta') v_t(\theta') \mathrm{d}\mu^+(\theta') + O(\epsilon)$$

(c.f., Otto calculus)

→ Negative curvature direction exponentially grows up!

→ μ_t moves away from the critical point.



Theorem (Informal)

The solution is not captured by any critical point *almost surely*.
(The solution converges to the global optimal solution almost surely)

Decay speed of objective

Suppose that $\left\|\frac{\mathrm{d}\mu^\circ}{\mathrm{d}\mu_t}\right\|_\infty \leq R$ (which could be ensured by using birth-death process).

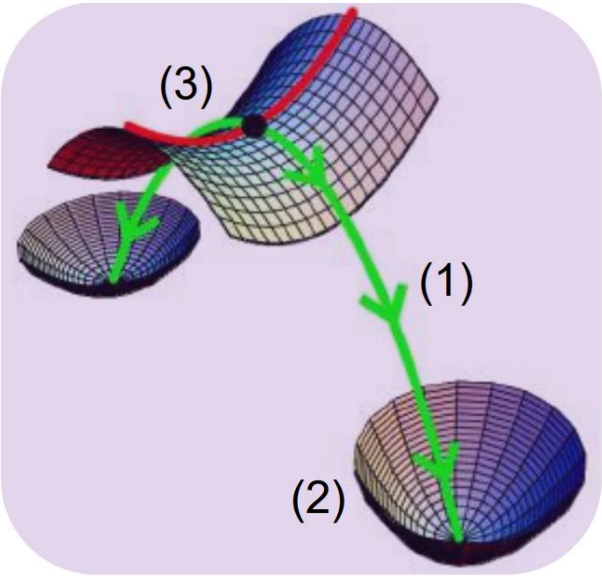
Theorem (GF moves toward a descent direction (1))

$$\frac{\mathrm{d}}{\mathrm{d}s}\mathcal{L}(\bar{\mu}_s)\Big|_{s=0} < -\delta \quad \Rightarrow \quad \frac{\mathrm{d}}{\mathrm{d}t}\mathcal{L}(\mu_t) \leq -R^{-1}\delta^2.$$

Theorem (Accelerated convergence phase (2))

Once $\mathcal{L}(\mu_t) \leq \frac{r^\circ}{2} - \epsilon$ is satisfied,

$$\mathcal{L}(\mu_{t+T}) \leq O\left(\frac{Rk^2}{T}\right)$$



Theorem (Negative curvature around a saddle point (3))

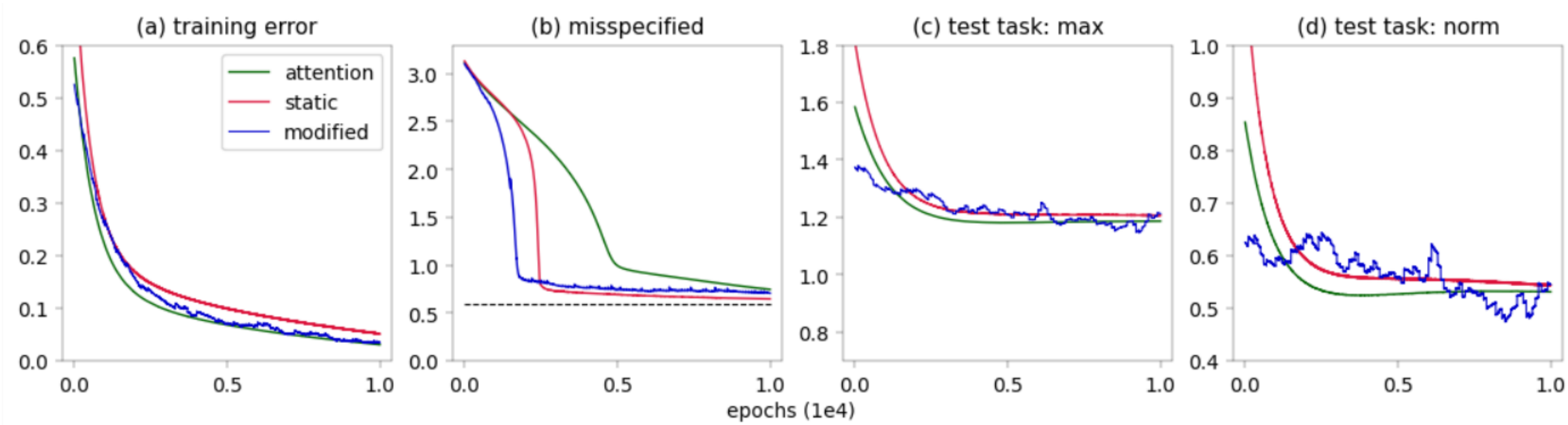
$$\frac{\mathrm{d}^2\mathcal{L}(\bar{\mu}_s)}{\mathrm{d}s^2} \leq -\Lambda \quad \Rightarrow \quad \text{min-eigen-value}(H_{\mu_t}) \leq -\Lambda/R$$

➡ Escape from the critical point exponentially fast.

Numerical experiment

We compare 3 models with $d = 20$, $k = 5$, and 500 neurons with sigmoid act.
All models are pre-trained using SGD on 10K prompts of 1K token pairs.

1. **attention**: jointly optimizes $\mathcal{L}(\mu, \Gamma)$.
2. **static**: directly minimizes $\mathcal{L}(\mu)$.
3. **modified**: static model implementing birth-death & GP



→ verify global convergence as well as improvement for misaligned model ($k_{\text{true}} = 7$) and nonlinear test tasks $g(x) = \max_{j \le k} h_{\mu^\circ}(x)_j$ or $g(x) = \|h_{\mu^\circ}(x)\|^2$.

Presentation overview

Statistics

Minimax optimality

- Nonparametric analysis
- Approximation error analysis

Optimization

Global optimality of nonlinear feature learning

- Mean field limit
- Strict saddle

Statistics/Optimization

Feature learning with one step GD

- Single index model
- Information exponent
- Advantage of pre-training

- [Minimax optimality and approximation error bound] Kim, Nakamaki, Suzuki: Transformers are Minimax Optimal Nonparametric In-Context Learners. NeurIPS2024
- [Optimization in mean field limit] Kim, Suzuki: Transformers Learn Nonlinear Features In Context: Nonconvex Mean-field Dynamics on the Attention Landscape. ICML2024 (arXiv:2402.01258).
- [Identifying low dimensional subspace with information exponent k] Oko, Song, Suzuki, Wu: Transformer efficiently learns low-dimensional functions in context. NeurIPS2024.

Nonlinear feature learning with optimization guarantee

[Oko, Song, Suzuki, Wu: Transformer efficiently learns low-dimensional functions in context. NeurIPS2024]

Kazusato Oko
(The University of Tokyo/RIKEN-AIP)

Yujin Song
(The University of Tokyo)

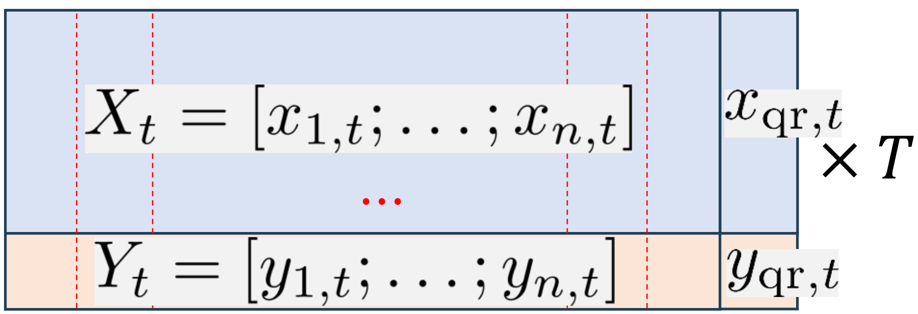
Denny Wu
(NYU/Flatiron Institute)

Mathematical formulation of in-context learning

Model: $y_{i,t} = f_*^t(x_{i,t}) + \epsilon_{i,t} \qquad (i = 1, \ldots, n)$

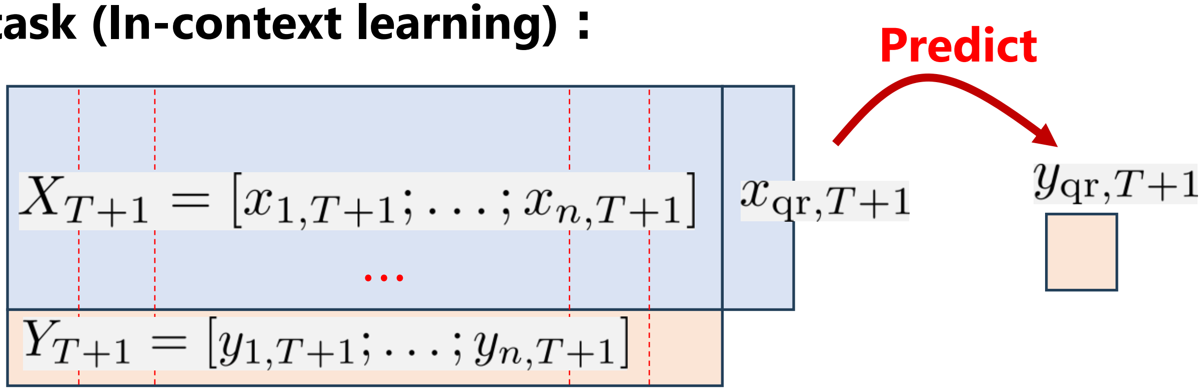
$t = 1, \ldots, T$: Task index

Pretraining (T tasks)：



- We observe pretraining task data T times.
- Each task has n data.

Test task (In-context learning)：



Teacher model

Gaussian single index model:

$$f_*^t(x) = \sigma_*^t(\langle x, \beta_t \rangle)$$

where the link σ_*^t and the direction β_t are generated randomly:

$\boldsymbol{\beta_t}$ β_t is distributed uniformly on a unit sphere in an $r < d$ dimensional linear subspace $\mathcal{S}$:

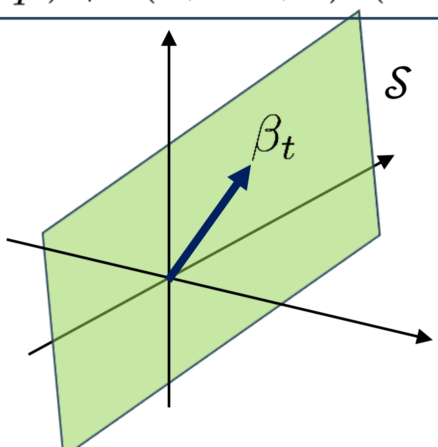
$$\beta_t \sim \mathrm{Unif}(\mathrm{Unit}(\mathcal{S})) \quad \text{where} \quad \dim(\mathcal{S}) = r \ll d$$

$\boldsymbol{\sigma_*^t}$ $\sigma_*^t(z) = \sum_{i=k}^{P} c_i^t \mathrm{He}_i(z)$

where c_i^t is randomly generated from a distribution satisfying

$$\mathbb{E}[c_2^t] \neq 0, \ \sum_{i=2}^{P} (c_i^t)^2 = \Theta(1) \ (\text{a.s.}), \ (c_2^t, \ldots, c_P^t) \neq (0, \ldots, 0) \ (\text{a.s.})$$

$\Rightarrow$ **Information exponent = $\boldsymbol{k}$.**



The feature has a low dimensional structure.

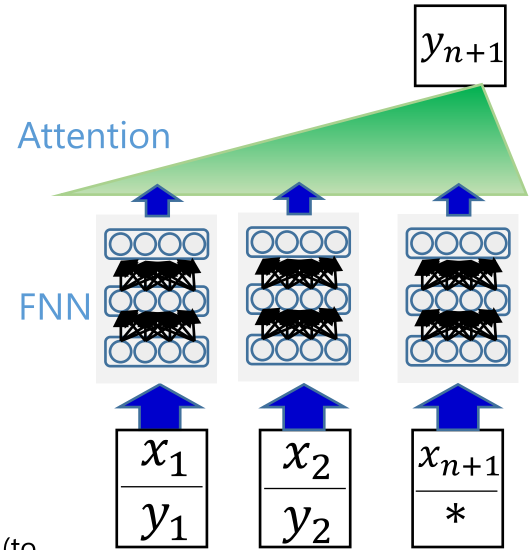
We want to estimate the subspace $\mathcal{S}$ and the basis functions He_i in the pretraining stage.

(Linear) Attention

- **FNN layer** $(f_W : \mathbb{R}^d \to \mathbb{R}^m)$ **:**

$$f_{\mathbf{W},\mathbf{b}}(x) = \begin{pmatrix} \sigma(\mathbf{w}_1^\top x + b_1) \\ \sigma(\mathbf{w}_2^\top x + b_2) \\ \vdots \\ \sigma(\mathbf{w}_m^\top x + b_m) \end{pmatrix} =: \sigma(\mathbf{W}x + \mathbf{b})$$

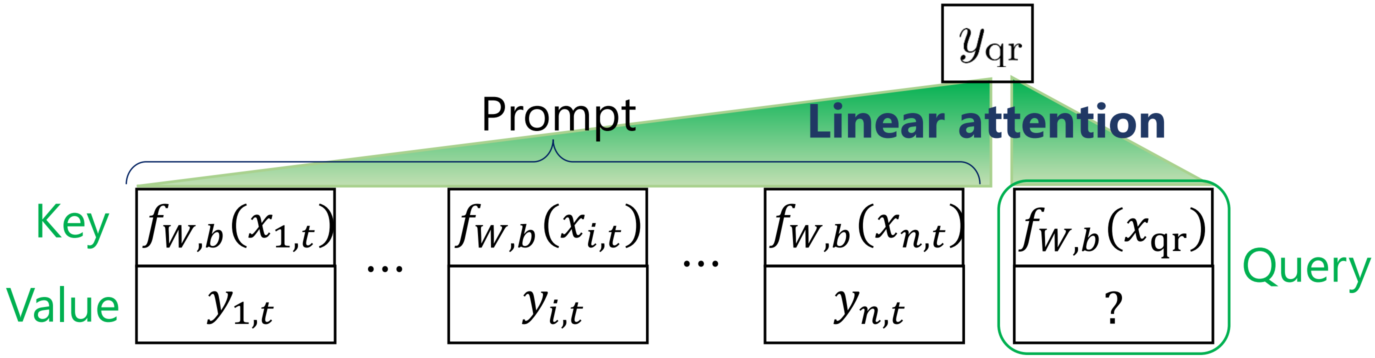
(σ: ReLU)



- **Linear attention model:** [Ahn et al.: Linear attention is (maybe) all you need (to understand transformer optimization). arXiv:2310.01082]

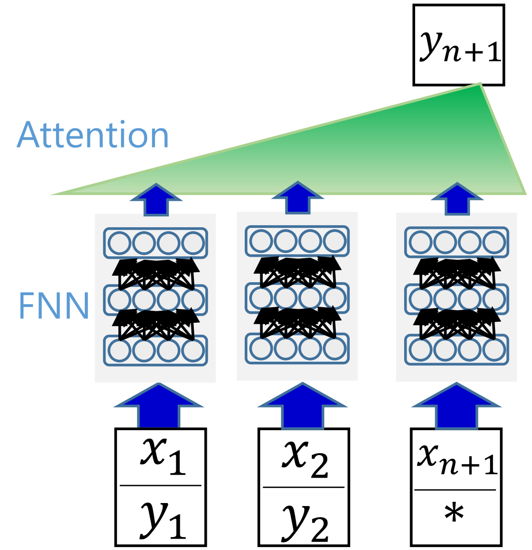
$$f(X_t, Y_t, x; \mathbf{W}, \mathbf{b}, \Gamma) = \frac{1}{n}\sum_{i=1}^{n} \underbrace{y_{i,t}}_{\text{Value}} \underbrace{f_{\mathbf{W},\mathbf{b}}(x_{i,t})^\top}_{\text{Key}} \Gamma \underbrace{f_{\mathbf{W},\mathbf{b}}(x_{\mathrm{qr}})}_{\text{Query}} \xrightarrow{\text{Predict}} y_{\mathrm{qr}} \qquad (\Gamma \in \mathbb{R}^{k\times k})$$

(linear regression)



Connection to soft-max attention

$$E = \begin{pmatrix} \sigma(\mathbf{w}_1^\top x_1 + b_1) & \dots & \sigma(\mathbf{w}_1^\top x_n + b_1) & \sigma(\mathbf{w}_1^\top x_{n+1} + b_1) \\ \vdots & \ddots & \vdots & \vdots \\ \sigma(\mathbf{w}_m^\top x_1 + b_m) & \dots & \sigma(\mathbf{w}_m^\top x_n + b_m) & \sigma(\mathbf{w}_m^\top x_{n+1} + b_m) \\ y_1 & \dots & y_n & 0 \end{pmatrix}$$



$$\begin{aligned} f_{\mathrm{Att}}(X,Y) &= W_V E \cdot \mathrm{softmax}\left(\frac{(W_K E)^\top W^Q E}{\lambda}\right) \\ &= \frac{1}{C_{n+1}} \sum_{j=1}^{n} (W_V E_{:,j}) \exp\left(\frac{(W_K E_{:,j})^\top (W_Q E_{:,n+1})}{\lambda}\right) \end{aligned}$$

Consider the following special setting:

$$W_V = \begin{bmatrix} 0_{1\times m} & 1 \end{bmatrix} \qquad W_K^\top W_Q = \begin{pmatrix} \Gamma & * \\ 0_{1\times m} & * \end{pmatrix}$$

Then,

$$f_{\mathrm{Att}}(X,Y) = \frac{1}{\cancel{C_{n+1}}} \sum_{j=1}^{n} y_j \cancel{\exp} \left(f_{\mathbf{w},\mathbf{b}}(x_j)^\top \Gamma f_{\mathbf{w},\mathbf{b}}(x_{n+1})\right)$$

By ignoring the normalization constant C_{n+1} and the nonlinear term exp, we obtain the linear attention in the previous slide.

In-Context Learning (ICL) risk

Empirical ICL risk :

$$\widehat{\mathcal{L}}(\mathbf{W},\mathbf{b},\Gamma) := \frac{1}{T}\sum_{t=1}^{T}\left(y_{\mathrm{qr},t} - \frac{1}{n}\sum_{i=1}^{n} y_{i,t} f_{\mathbf{W},\mathbf{b}}(x_{i,t})^\top \Gamma f_{\mathbf{W},\mathbf{b}}(x_{\mathrm{qr},t})\right)^2$$

→ Minimize with respect to W, b, Γ.

The expected ICL risk: (Large sample limit: $n \to \infty$ and $T \to \infty$)

$$\mathcal{L}(\mathbf{W},\mathbf{b},\Gamma) := \mathbb{E}_{x_{\mathrm{qr}},f_*}\left[\left(f_*(x_{\mathrm{qr}}) - \mathbb{E}_x[f_*(x) f_{\mathbf{W},\mathbf{b}}(x)^\top]\Gamma f_{\mathbf{W},\mathbf{b}}(x_{qr})\right)^2\right]$$

(note that $y_{i,t} = f_*^t(x_{i,t}) + \epsilon_{i,t}$)

Question :

- Can we estimate W, b, Γ by gradient descent? (Non-convex problem)
- How large is the sample complexity?

Optimization algorithm

Initialize $\boldsymbol{w}_j^{(0)} \sim \mathrm{Unif}(\mathbb{S}^{d-1})$, $b_j = 0$, $\Gamma_{j,j}^{(0)} = \mathrm{Unif}(\{\pm 1\})$ (diagonal).

- **Stage 1: One-step gradient descent.**

Optimize $\boldsymbol{W}$ by a **one-step gradient descent**:

Find the subspace $\mathcal{S}$

$$\mathbf{w}_j^{(1)} \leftarrow \mathbf{w}_j^{(0)} - \eta \left[\nabla_{\mathbf{w}_j} \frac{1}{T_1} \sum_{t=1}^{T_1} \left(y_{\mathrm{qr},t} - f(X_t, Y_t, x_{\mathrm{qr},t}; \mathbf{W}^{(0)}, \mathbf{b} = 0, \Gamma^{(0)}) \right)^2 + \lambda \mathbf{w}_j^{(0)} \right]$$

- ➢ Analogous to one-step GD for 2-layer NN [Damian et al. 22; Ba et al. 22].
- ➢ Since the true link function has $\mathrm{IE} = 2$, we can recover the subspace $\mathcal{S}$ by one-step GD with large step size.

- **Stage 2: Optimization of Γ.**

Randomly re-initialize $b_j \sim \mathrm{Unif}([-1,1])$.

Optimize Γ based on the feature $\boldsymbol{W}$ obtained at Stage 1:

$$\widehat{\Gamma} \leftarrow \arg\min_{\Gamma} \left\{ \frac{1}{T_2} \sum_{t=T_1+1}^{T_1+T_2} \left(y_{\mathrm{qr},t} - f(X_t, Y_t, x_{\mathrm{qr},t}; \mathbf{W}^{(1)}, \mathbf{b}, \Gamma) \right)^2 + \lambda \|\Gamma\|_F^2 \right\}$$

$$\frac{1}{n} \sum_{i=1}^{n} y_{i,t} f_{\mathbf{W},\mathbf{b}}(x_{i,t})^\top \Gamma f_{\mathbf{W},\mathbf{b}}(x_{\mathrm{qr}})$$

Train the attention to extract the coefficient β_t.

Stage 2

$$\widehat{\Gamma} \leftarrow \arg\min_{\Gamma} \left\{ \frac{1}{T_2} \sum_{t=T_1+1}^{T_1+T_2} \left(y_{\mathrm{qr},t} - f(X_t, Y_t, x_{\mathrm{qr},t}; \mathbf{W}^{(1)}, \mathbf{b}, \Gamma) \right)^2 + \lambda \|\Gamma\|_F^2 \right\}$$

$$f_t(X_t, Y_t, x_{\mathrm{qr},t}; \mathbf{W}, \mathbf{b}, \Gamma) = \frac{1}{n} \sum_{i=1}^{n} y_{i,t} f_{\mathbf{W},\mathbf{b}}(x_{i,t})^\top \Gamma f_{\mathbf{W},\mathbf{b}}(x_{\mathrm{qr},t})$$

Then, $\hat{\Gamma}$ performs the ridge regression:

$$f_t(X_t, Y_t, x_{\mathrm{qr},t}; \mathbf{W}^{(1)}, \mathbf{b}, \hat{\Gamma}) = f_{\mathbf{W}^{(1)},\mathbf{b}}(x_{\mathrm{qr},t})^\top \left(\frac{1}{nT_2} F_{T_1:T_2}^\top F_{T_1:T_2} + \lambda I \right)^{-1} F_t Y_t$$

where $F_t = [f_{\mathbf{W}^{(1)},\mathbf{b}}(x_{1,t}), \ldots, f_{\mathbf{W}^{(1)},\mathbf{b}}(x_{n,t})]$.

If we can obtain **nice basis functions** $f_{\boldsymbol{W}^{(1)},\boldsymbol{b}}$ at Stage 1, the target function can be well estimated in the test task.

Main result

Theorem (ICL risk bound)

Let n^* be the number of examples in test task. If the one-step GD is performed with

$$T_1 = \Theta(d^{k+1}) \text{ and } n = \widetilde{\Omega}(d^k),$$

then the trained Transformer achieves the following test loss:

$$\mathcal{L}(\widehat{\mathbf{W}}, \widehat{\mathbf{b}}, \widehat{\Gamma}) \lesssim \underbrace{\frac{r^{3P/2}}{\sqrt{m}}}_{\text{Approximation error}} + \underbrace{\sqrt{\frac{r^{4P}}{T_2}}}_{\text{Error to estimate } \Gamma} + \underbrace{r^{2P}\sqrt{\frac{1}{n} + \frac{1}{n^*}}}_{\text{Error to estimate in the test task}}.$$

m: width of NN, T_1: number of tasks in Stage 1 (learning W), T_2: number of tasks in Stage 2 (learning Γ), n: number of examples in pretraining-task.

- Without pretraining (non-ICL setting), $n^* = \Omega(d^p)$ for kernel method and $n^* = \Omega(d^{k/2})$ for CSQ algorithm are required. But, in ICL, n^* can be independent of d ($n^* = \text{poly}\ (r)$).
- To estimate W, it requires $T_1 n = \Theta(d^{2k+1})$ datapoints while Damian et al. (2022) required only $\Theta(d^2)$ data points because we need enough task diversity.
  - ➢ But, ICL does not update their parameters based on the in-context examples.

Main result

Theorem (ICL risk bound)

Let n^* be the number of examples in test task. If the one-step GD is performed with

$$T_1 = \Theta(d^{k+1}) \text{ and } n = \tilde{\Omega}(d^k),$$

then the trained Transformer achieves the following test loss:

$$\mathcal{L}(\widehat{\mathbf{W}}, \hat{\mathbf{b}}, \hat{\Gamma}) \lesssim \underbrace{\frac{r^{3P/2}}{\sqrt{m}}}_{\text{Approximation error}} + \underbrace{\sqrt{\frac{r^{4P}}{T_2}}}_{\text{Error to estimate } \Gamma} + \underbrace{r^{2P}\sqrt{\frac{1}{n} + \frac{1}{n^*}}}_{\text{Error to estimate in the test task}}.$$

m: ... 2

(lea

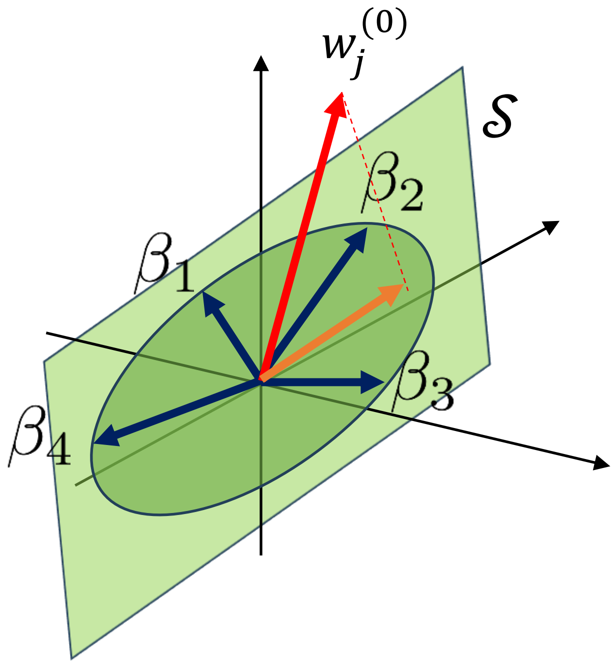
	w/o pretraining		w/ pretraining
Method	Kernel	NN (CSQ or SQ)	ICL
Sample complexity	d^P	$d^{k/2}$ (or d)	r^{2P}
Pretraining	---	---	$T_1 = d^{k+1}, n = d^k$

- V ... Ω ... d
- To esti ... al. (2022) require ... ty.
  - Bu ... t examples.

**If we observe many data during pretraining,
ICL with Transformer can generalize well in test tasks.**

Proof overview

- The one-step GD update (with regularization) projects the initial vector $w_j^{(0)}$ to the subspace $\mathcal{S}$.



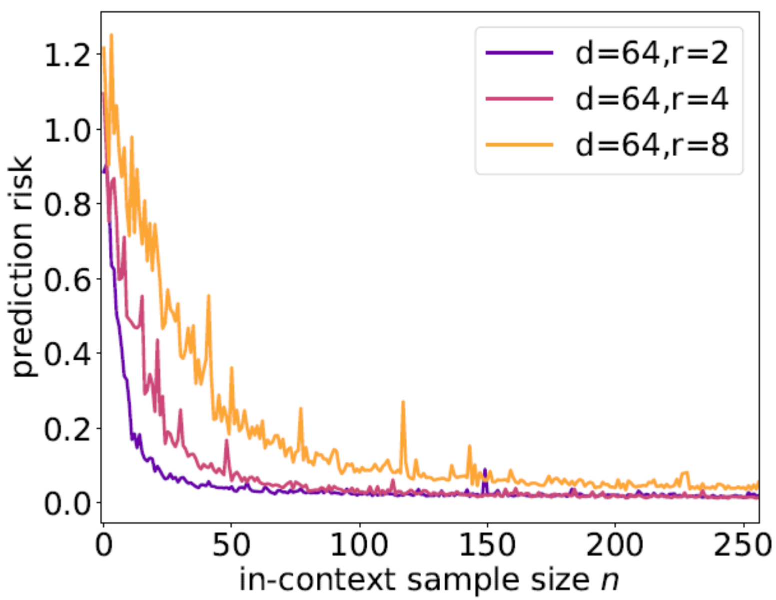
- Learning W : Subspace $\mathcal{S}$ is obtained.
- Learning Γ : Attention to obtain the coefficients on basises.

- If we have many neurons, $\left(w_j^{(1)}\right)_{j=1}^{m}$ spans the subspace $\mathcal{S}$ (**1st -stage**).

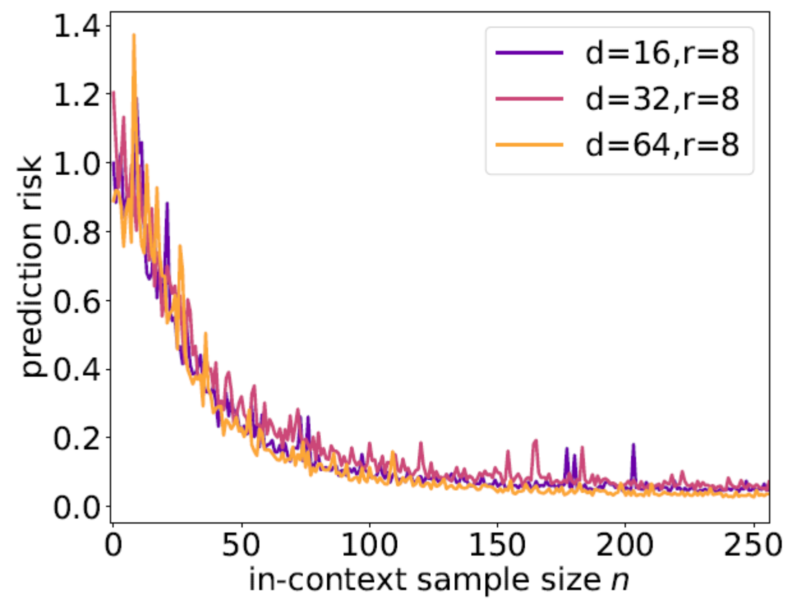
- If we have sufficiently large number of neuros $\left(\sigma(w_j^{(1)\top} w + b_j)\right)_{j=1}^{m}$, the model can well approximate the target polynomial $\sigma^*(\langle \beta_t, x \rangle)$ by **linear combination of the ReLU-neurons (2nd-stage + test prompt)**.

Numerical experiment

Fixing d, changing r



Fixing r, changing d



GPT2 model with 12-layers (~22M parameters)

Only r affects the result, d does not.

Summary

- Learning theory of in-context learning

$$F_t^\circ(x) = \beta_t^\top f^\circ(x)$$

  - **Pretraining:** Obtaining nonlinear feature [f°]
  - **In-context instruction:** Learning coefficient [β_t]

- Nonparametric regression theory
  - Minimax optimality
  - Task diversity matters.
- Optimization theory
  - Feature learning by mean-field neural network
  - Estimating single-index model by gradient descent

  → Feature learning helps to improve the sample complexity of in-context learning.