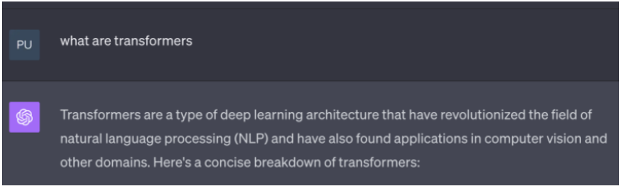
In-Context Convergence of Transformers

Yu Huang[1] Yuan Cheng[2] Yingbin Liang[3]

[1] *University of Pennsylvania*
[2] *National University of Singapore*
[3] *The Ohio State University*

Transformers have revolutionized many domains

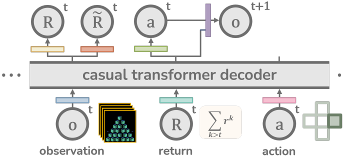
Attention is all you need. Vaswani et al, (2017)

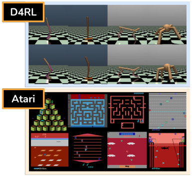


Large Language Models (LLMs)

Computer Vision

DALL-E3





Reinforcement Learning

Wu et al. (2023)

Remarkable emergent ability for LLMs: In-Context learning

Given a prompt containing in-context examples, pre-trained LLM responds to new query token appropriately *without further fine-tuning*.

$$\underbrace{\text{maison} \rightarrow \text{house, chat} \rightarrow \text{cat, chien} \rightarrow}_{\text{prompt}} \quad \underbrace{\text{dog}}_{\text{completion}}$$

French
$\Downarrow$
English

This figure is from Garg et al., (2022)

Remarkable emergent ability for LLMs: In-Context learning

Given a prompt containing in-context examples, pre-trained LLM responds to new query token appropriately *without further fine-tuning*.

$$\underbrace{\text{maison} \to \text{house},\ \text{chat} \to \text{cat},\ \text{chien} \to}_{\text{prompt}}\ \underbrace{\text{dog}}_{\text{completion}}$$

French
⇓
English

This figure is from Garg et al., (2022)

Learning **without** **weight updates**

Towards understanding ICL with transformers

Empirical evidence

- Transformers can in-context learn **linear** function (Garg et al, 2022)
  - Sample many $f \in \mathcal{F}$, and construct corresponding prompts:

$$\left(x_1, f\left(x_1\right), \ldots, x_N, f\left(x_N\right), x_{\text{query}}\right)$$

  - Train transformer to predict $f(x_{\text{query}})$
  - For a new f' and its prompt: the trained model (without finetuning) can predict $f'(x_{\text{query}})$

Towards understanding ICL with transformers

Empirical evidence

- Transformers can in-context learn **linear** function (Garg et al, 2022)
  - Sample many $f \in \mathcal{F}$, and construct corresponding prompts:

$$(x_1, f(x_1), \ldots, x_N, f(x_N), x_{\text{query}})$$

  - Train transformer to predict $f(x_{\text{query}})$
  - For a new f' and its prompt: the trained model (without finetuning) can predict $f'(x_{\text{query}})$

Going forward theoretically

- Most focus on Expressive power or Generalization:
  Oswald et al., (2023); Bai et al., (2023); Li et al., (2023)

Towards understanding ICL with transformers

Empirical evidence

- Transformers can in-context learn **linear** function (Garg et al, 2022)
  - Sample many $f \in \mathcal{F}$, and construct corresponding prompts:

$$(x_1, f(x_1), \ldots, x_N, f(x_N), x_{\text{query}})$$

  - Train transformer to predict $f(x_{\text{query}})$
  - For a new f' and its prompt: the trained model (without finetuning) can predict $f'(x_{\text{query}})$

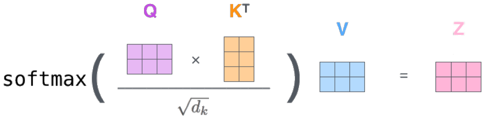
Going forward theoretically

- Most focus on Expressive power or Generalization:
  Oswald et al., (2023); Bai et al., (2023); Li et al., (2023)

- Training dynamics of **linear** attention:
  Zhang et al., (2023); Mahankali et al., (2023); Ahn et al., (2023)

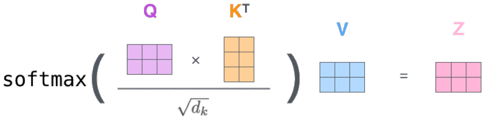
How do softmax-based transformers trained via gradient descent learn in-context?

- Trasnformers are based on **softmax** attention mechanism.



Our contribution: first step towards in-context learning dynamics of the 1-layer softmax transformer

- Trasnformers are based on **softmax** attention mechanism.



ICL framework

- **Prompt:** $P = (x_1, f(x_1), \ldots, x_N, f(x_N), x_{\text{query}})$:
  - **Linear task:** $f(x) = \langle w, x \rangle$, $w \sim \mathcal{D}_\Omega$
  - **IID data:** $\{x_i\} \cup \{x_{\text{query}}\} \overset{i.i.d}{\sim} D_{\mathcal{X}}$
- **Goal:** Predict $\widehat{y}_{\text{query}} \approx f(x_{\text{query}})$.

ICL framework

- **Prompt:** $P = (x_1, f(x_1), \ldots, x_N, f(x_N), x_{\text{query}})$:
  - **Linear task:** $f(x) = \langle w, x \rangle$, $w \sim \mathcal{D}_\Omega$
  - **IID data:** $\{x_i\} \cup \{x_{\text{query}}\} \overset{i.i.d}{\sim} D_{\mathcal{X}}$
- **Goal:** Predict $\widehat{y}_{\text{query}} \approx f(x_{\text{query}})$.

- **Task Distribution $\mathcal{D}_\Omega$:**
  - $w \overset{i.i.d}{\sim} \mathcal{D}_\Omega$ with zero mean and covariance $\mathbf{I}_{d \times d}$

ICL framework

- **Prompt:** $P = (x_1, f(x_1), \ldots, x_N, f(x_N), x_{\text{query}})$:
  - **Linear task:** $f(x) = \langle w, x\rangle$, $w \sim \mathcal{D}_\Omega$
  - **IID data:** $\{x_i\} \cup \{x_{\text{query}}\} \overset{i.i.d}{\sim} D_{\mathcal{X}}$
- **Goal:** Predict $\widehat{y}_{\text{query}} \approx f(x_{\text{query}})$.

- **Task Distribution $\mathcal{D}_\Omega$:**
  - $w \overset{i.i.d}{\sim} \mathcal{D}_\Omega$ with zero mean and covariance $\mathbf{I}_{d\times d}$
- **Data Distribution $\mathcal{D}_{\mathcal{X}}$:**
  - K distinct features:

$$v_k \in \mathbb{R}^d, \|v_k\| = 1 \text{ for } k \in [K], v_i \perp v_j \text{ for } i \neq j$$

  - $x = v_k$ with prob p_k, where $p_k \in (0,1)$ and $\sum_{k\in[K]} p_k = 1$.

Transformer Architecture

- **Embeddings**

$$E = E(P) = \begin{pmatrix} x_1 & x_2 & \cdots & x_N & x_{\text{query}} \\ y_1 & y_2 & \cdots & y_N & 0 \end{pmatrix} \begin{array}{l} E^x \\ E^y \end{array} \in \mathbb{R}^{(d+1)\times(N+1)}.$$

Transformer Architecture

- **Embeddings**

E^{x}

$$E = E(P) = \begin{pmatrix} x_1 & x_2 & \cdots & x_N & x_{\text{query}} \\ y_1 & y_2 & \cdots & y_N & 0 \end{pmatrix} \in \mathbb{R}^{(d+1)\times(N+1)}.$$

E^{y}

- **One-layer transformer:**
  - Self-attention mechanism: $W^V E \cdot \mathrm{softmax}\left(\left(W^K E\right)^\top W^Q E\right)$,
  - Mask: $W^V M(E)$, $W^K M(E)$
  - Reparameterization: $\theta = (\nu, Q)$ (Anh et al., 2023, Zhang et al., 2023)

$$W^V = \begin{pmatrix} 0_{d\times d} & 0_d \\ 0_d^\top & \nu \end{pmatrix}, \quad W^{KQ} = \begin{pmatrix} Q & 0_d \\ 0_d^\top & 0 \end{pmatrix}$$

- Consolidate Key and Query
- Fixed $\nu = 1$

\- Nealy no loss of optimality!

Transformer Architecture

- **Output:**

$$\widehat{y}^{(t)}_{\mathsf{query}} = [M(E^y) \cdot \mathrm{softmax}\left(M(E^x)^\top Q^{(t)} E^x\right)]_{N+1}$$
$$= \sum_{i \in [N]} \mathbf{attn}_i^{(t)} y_i = \sum_{k \in [K]} \mathbf{Attn}_k^{(t)} \langle w, v_k \rangle.$$

Transformer Architecture

- **Output:**

$$\widehat{y}^{(t)}_{\mathsf{query}} = [M(E^y) \cdot \mathrm{softmax}\left(M(E^x)^\top Q^{(t)} E^x\right)]_{N+1}$$
$$= \sum_{i \in [N]} \mathbf{attn}_i^{(t)} y_i = \sum_{k \in [K]} \mathbf{Attn}_k^{(t)} \langle w, v_k \rangle.$$

- for the i-th token: $\mathbf{attn}_i^{(t)} = \frac{e^{E_i^{x\top} Q^{(t)} E_{N+1}^x}}{\sum_{j \in [N]} e^{E_j^{x\top} Q^{(t)} E_{N+1}^x}}$.
- for the k-th features: $\mathbf{Attn}_k^{(t)} = \sum_{i \in [N], x_i = v_k} \mathbf{attn}_i^{(t)}$.

Training Settings

- **Loss Function:**

$$L(\theta) = \frac{1}{2}\mathbb{E}_{w\sim\mathcal{D}_\Omega, \{x_i\}_{i=1}^N \cup \{x_{\text{query}}\}\sim\mathcal{D}_\mathcal{X}^{N+1}}\left[\left(\widehat{y}_{\text{query}} - \langle w, x_{\text{query}}\rangle\right)^2\right]$$

- **Training Algorithm:** $Q^{(0)}$ initialize as $\mathbf{0}_{d\times d}$, with GD update.
- **Prediction Error:**

$$\mathcal{L}_k(\theta) = \frac{1}{2}\mathbb{E}\left[\left(\widehat{y}_{\text{query}} - \langle w, x_{\text{query}}\rangle\right)^2 \middle| x_{\text{query}} = v_k\right].$$

performance measure

ICL with imbalanced features

Imbalanced Cases: One dominant feature v_1: $p_1 = \Theta(1)$;
Under-represented v_k: $p_k = \Theta\left(\frac{1}{K}\right)$.

Theorem (Prediction Error Converges (Informal))

For $0 < \epsilon < 1$, $N \geq \mathrm{poly}(K)$, $\mathrm{polylog}(K) \gg \log(\frac{1}{\epsilon})$, *prediction error:*

1. **Dominant** *feature* v_1*: with at most* $T_1 = O(\frac{\log(\epsilon^{-1/2})}{\eta\epsilon})$ *GD iterations,* $\mathcal{L}_1(\theta^{(T_1)}) \leq \mathcal{L}_1^* + \epsilon$.
2. **Under-represented** *features* v_k*: with at most* $T_k = O(\frac{\log(K)K^2}{\eta} + \frac{K\log\left(K\epsilon^{-\frac{1}{2}}\right)}{\epsilon\eta})$ *GD iterations,* $\mathcal{L}_k(\theta^{(T_k)}) \leq \mathcal{L}_k^* + \epsilon$.

ICL with imbalanced features

Imbalanced Cases: One dominant feature v_1: $p_1 = \Theta(1)$;
Under-represented v_k: $p_k = \Theta\left(\frac{1}{K}\right)$.

Theorem (Prediction Error Converges (Informal))

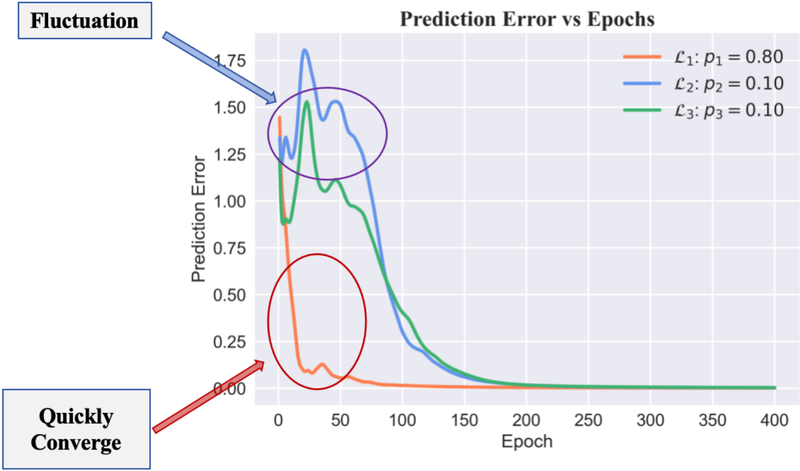
For $0 < \epsilon < 1$, $N \geq \mathrm{poly}(K)$, $\mathrm{polylog}(K) \gg \log(\frac{1}{\epsilon})$, *prediction error:*

1. **Dominant** *feature* v_1*: with at most* $T_1 = O(\frac{\log(\epsilon^{-1/2})}{\eta\epsilon})$ *GD iterations,* $\mathcal{L}_1(\theta^{(T_1)}) \leq \mathcal{L}_1^* + \epsilon$.
2. **Under-represented** *features* v_k*: with at most* $T_k = O(\frac{\log(K)K^2}{\eta} + \frac{K\log\left(K\epsilon^{-\frac{1}{2}}\right)}{\epsilon\eta})$ *GD iterations,* $\mathcal{L}_k(\theta^{(T_k)}) \leq \mathcal{L}_k^* + \epsilon$.

- Global optimal: $\mathcal{L}_k^* = \Theta(e^{-\mathrm{poly}(K)})$.
- Nearly optimal prediction error for both under-represented features and the dominant feature.
- **Stage-wise Convergence!**

ICL with imbalanced features

Stage-wise Convergence



ICL with imbalanced features

Theorem (Attention score concentrates (Informal))

For any $0 < \epsilon < 1$, $N \geq \mathrm{poly}(K)$, $\mathrm{polylog}(K) \gg \log(\frac{1}{\epsilon})$, *for attention score,* $x_{query} = v_k$, *after* T_k, *w.h.p*

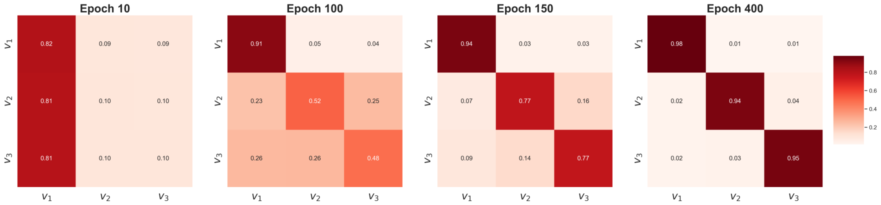
$$(1 - \mathbf{Attn}_k^{(T_k)})^2 \leq O(\epsilon).$$

ICL with imbalanced features

Theorem (Attention score concentrates (Informal))

For any $0 < \epsilon < 1$, $N \geq \mathrm{poly}(K)$, $\mathrm{polylog}(K) \gg \log(\frac{1}{\epsilon})$, *for attention score,* $x_{query} = v_k$, *after* T_k, *w.h.p*

$$(1 - \mathbf{Attn}_k^{(T_k)})^2 \leq O(\epsilon).$$



Attention scores Heatmap

ICL with imbalanced features

Theorem (Attention score concentrates (Informal))

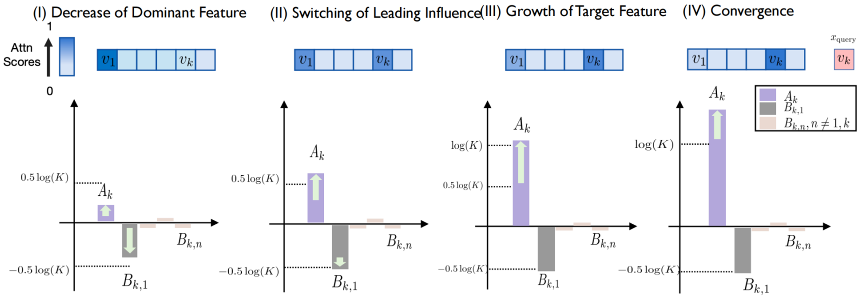
For any $0 < \epsilon < 1$, $N \geq \mathrm{poly}(K)$, $\mathrm{polylog}(K) \gg \log(\frac{1}{\epsilon})$, *for attention score,* $x_{\mathit{query}} = v_k$, *after* T_k, *w.h.p*

$$(1 - \mathbf{Attn}_k^{(T_k)})^2 \leq O(\epsilon).$$

In-context Ability: given a test prompt from any new task w (possibly unseen), model can still well approximate test query.

$$y_{\text{query}}^{(T^*)} = \mathbf{Attn}_k^{(T^*)} \langle w, v_k \rangle + \sum_{m \neq k} \mathbf{Attn}_m^{(T^*)} \langle w, v_m \rangle \approx \langle w, v_k \rangle.$$

Four-phase Behavior of Under-presented Features



Four-phase learning dynamics of under-represented features

Bilinear attention weight

A_k : weight of query token and its target feature

$B_{k,n}$: weight of query token and off-target features

Concluding remarks

- Analyzing the training dynamics of a one-layer transformer with **softmax** attention trained by GD for in-context learning.

Concluding remarks

- Analyzing the training dynamics of a one-layer transformer with **softmax** attention trained by GD for in-context learning.
- **Take away message**:
  - Stage-wise convergence
  - Attention concentration $\rightarrow$ In-context ability..
  - Novel analysis of phase decomposition.

Thanks & Questions