Theoretical Foundations of In-Context Learning and Chain-of-Thought Using Properly Trained Transformer Models

Presenter: Hongkang Li

Future of Computing Research Collaboration at Rensselaer

Development of deep learning

Take the area of NLP as an example.

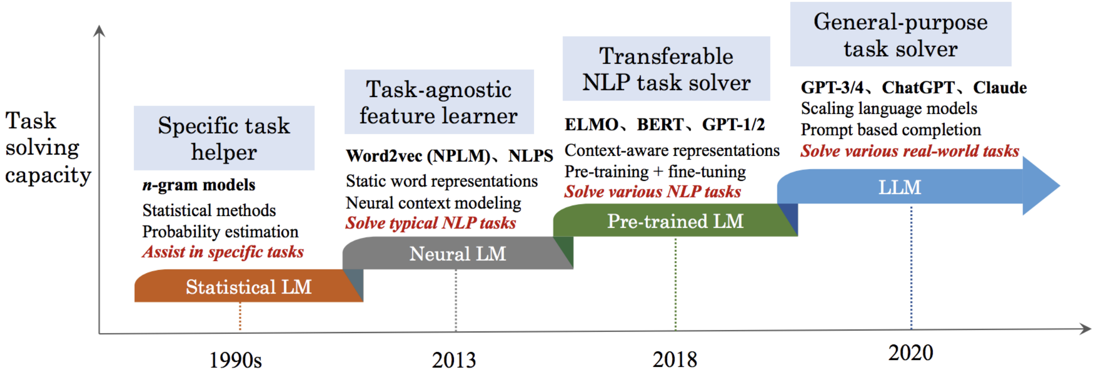


Figure 1: Deep Learning paradigm[1]

[1]source from [Zhao et al.23]

Large Language Model (LLM) and In-context learning (ICL)

- Transformer-based foundation models, e.g., ChatGPT, GPT-4, Sora, have achieved great empirical success in many areas.
- Large foundation models are able to implement **in-context learning (ICL)** and reasoning.

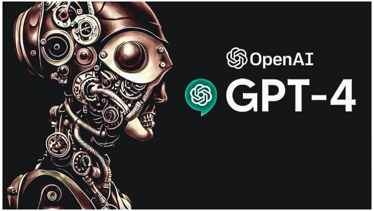


Figure 2: *GPT-4. Source from medium*

Figure 3: *Sora. Source from medium*

Large Language Model (LLM) and In-context learning (ICL)

- In-context learning makes predictions for new tasks on pre-trained LLM without fine-tuning the model.
- It is implemented by providing a few testing examples and necessary instructions as a prompt for the testing data.

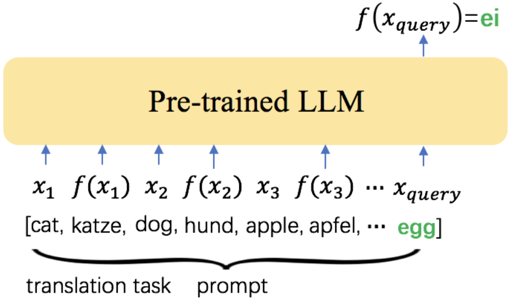


Figure 4: Machine Translation with ICL

Our focus

Despite the empirical success of ICL, one fundamental and theoretical question for ICL is less investigated, i.e.,

How can a Transformer be trained to perform ICL and generalize in and out of domain successfully and efficiently?

Specifically,

- What are the sufficient conditions for out-of-domain ICL?
- What is the mechanism of ICL?
- Can we prune the model in in-context inference and why?

Related works

[Garg et al.22, Akyurek et al. 23] propose a framework for studying ICL on linear regression.

- Consider a prompt $P = (x_1, f(x_1), x_2, f(x_2), \cdots, x_{query})$. f is a linear function.
- We say a model M can in-context learn f with up to an ϵ error to predict $f(x_{query})$, if

$$\mathbb{E}_P[\ell(M(P), f(x_{query}))] \leq \epsilon. \quad (1)$$

- The model M parameterized by Θ is trained by minimizing the risk function

$$\min_{\Theta} \mathbb{E}_{P,f}[\ell(M_\Theta(P^i), f(x^i_{query}))]. \quad (2)$$

- Results: the trained Transformer is able to learn unseen linear functions from in-context examples with performance comparable to the optimal least square estimator.

Related works

A few further works theoretically study the training dynamics and generalization of Transformers in implementing ICL.

- [Zhang et al.24, Wu et al.24] study linear regression tasks on $\{(x_n, f(x_n))\}_{n=1}^N$, where f is a linear function, using the prompt

$$P = \begin{pmatrix} x_1 & x_2 & \cdots & x_l & x_{query} \\ f(x_1) & f(x_2) & \cdots & f(x_l) & 0 \end{pmatrix} \in \mathbb{R}^{(d+1)\times(l+1)}. \tag{3}$$

  The training model they consider is a one-layer Transformer with linear attention,

$$F(P;\Theta) = P + W^{PV}P \cdot P^\top W^{KQ}P. \tag{4}$$

- [Zhang et al.24] further study the generalization when the data/task distribution shift exists; [Wu et al.24] characterize the required number of pretraining tasks for ICL.

Related works

Given the prompt in (3), [Huang et al.24] explore a one-layer Transformer with softmax attention on learning linear regression tasks, i.e.,

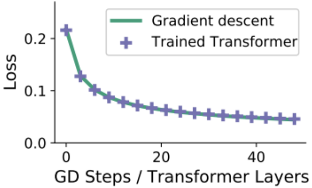
$$F(P;\Theta) = \sum_{i=1}^{N} y_i \mathrm{softmax}(x_i^\top \Theta x_{query}) \tag{5}$$

- [Huang et al.24] consider x_i as orthogonal features, following the line of feature-learning analysis.
- [Huang et al.24] in-depth characterize the dynamics of the training process under cases of balanced and imbalanced prompt examples.

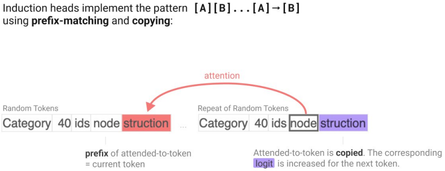
Related works

Some other works also study the **mechanism** of ICL implemented by Transformers.

Transformer=GD: [von Oswald et al.23] finds that a one-layer Transformer can implement one-step gradient descent via in-context inference. Further works [Ahn et el.23, Cheng et al.24] extend the conclusion to preconditioned GD and functional GD given different settings.



Induction head [Olsson et al.22]: Transformers find the answer from the prefix to generate the next token.



Our work and major contributions

Our recent work "How Do Nonlinear Transformers Learn and Generalize in In-Context Learning?"[2] at ICML 2024 has the following contributions.

- A theoretical characterization of how to train Transformers with nonlinear attention and nonlinear MLP and to enhance their ICL capability.
- Expand the theoretical understanding of the mechanism of the ICL capability of Transformers.
- Theoretical justification of Magnitude-based Pruning in preserving ICL.

[2] https://arxiv.org/pdf/2402.15607.pdf

Our work and major contributions

Summary of contributions and comparisons with related theoretical works.

Theoretical Works	Nonlinear Attention	Nonlinear MLP	Training Analysis	Distribution -Shifted Data	Tasks
[Zhang et al.24]			✓	✓	linear regression
[Huang et al.24]	✓		✓		linear regression
[Wu et al.24]			✓		linear regression
Ours	✓	✓	✓	✓	classification

Table 1: Comparison with existing works about training analysis and generalization guarantee of ICL

Problem formulation

We study binary classification problems. Given the input $\boldsymbol{x}_{query}$, we aim to predict the label $f(\boldsymbol{x}_{query})$ for the task f. We conduct training with constructed prompts $\boldsymbol{P}$ on a model to enable ICL.

$$\boldsymbol{P} = \begin{pmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_l & \boldsymbol{x}_{query} \\ \boldsymbol{y}_1 & \boldsymbol{y}_2 & \cdots & \boldsymbol{y}_l & 0 \end{pmatrix} := (\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_{query}). \tag{6}$$

- $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ are context inputs and outputs, respectively.
- $\boldsymbol{y}_i = embedding(f(\boldsymbol{x}_i))$ is an embedding of $f(\boldsymbol{x}_i)$. $\boldsymbol{y}_i = \boldsymbol{q}$ if $f(\boldsymbol{x}_i) = +1$. $\boldsymbol{y}_i = -\boldsymbol{q}$ if $f(\boldsymbol{x}_i) = -1$.
- We also name the parts of $\boldsymbol{x}$ and $\boldsymbol{y}$ as feature embedding and label embedding in $\boldsymbol{P}$, respectively

Problem formulation

Learning model: a single-head, one-layer Transformer with a self-attention layer and a two-layer perceptron, i.e.,

$$F(\Psi; \boldsymbol{P}) = \boldsymbol{a}^\top \mathrm{Relu}(\boldsymbol{W}_O \sum_{i=1}^{l} \boldsymbol{W}_V \boldsymbol{p}_i \cdot \mathrm{attn}(\Psi; \boldsymbol{P}, i)),$$
$$\mathrm{attn}(\Psi; \boldsymbol{P}, i) = \mathrm{softmax}((\boldsymbol{W}_K \boldsymbol{p}_i)^\top \boldsymbol{W}_Q \boldsymbol{p}_{query}) \tag{7}$$

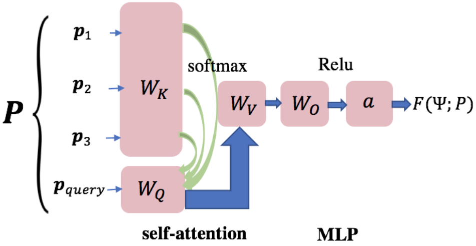


Figure 5: The Transformer network for learning

Problem formulation

Model training: The training is to solve the empirical risk minimization using N pairs of prompt and labels $\{\boldsymbol{P}^n, z^n\}_{n=1}^N$, $\Psi = \{\boldsymbol{W}_Q, \boldsymbol{W}_K, \boldsymbol{W}_V, \boldsymbol{W}_O, \boldsymbol{a}\}$,

$$\min_{\Psi} R_N(\Psi) := \frac{1}{N}\sum_{n=1}^{N} \ell(\Psi; \boldsymbol{P}^n, z^n) \tag{8}$$

- The query and context inputs are sampled from a distribution $\mathcal{D}$.
- The task f^n is sampled from a distribution $\mathcal{T}$. The training tasks form a set $\mathcal{T}_{tr} \subset \mathcal{T}$.
- $\ell(\Psi; \boldsymbol{P}^n, z^n) = \max\{0, 1 - z^n \cdot F(\Psi, \boldsymbol{P}^n)\}$ is the Hinge loss.
- The model is trained via stochastic gradient descent (SGD).
- $\boldsymbol{W}_Q$, $\boldsymbol{W}_K$, and $\boldsymbol{W}_V$ initialized from a small scaling of identity matrices. $\boldsymbol{W}_O$ initialized from Gaussian distribution.

Problem formulation

Generalization: We introduce in-domain and out-of-domain generalization.

- In-domain generalization: No distribution shift between training and testing data. The generalization error is defined as

$$\mathop{\mathbb{E}}_{\boldsymbol{x}_{query} \sim \mathcal{D}, f \in \mathcal{T} \setminus \mathcal{T}_{tr}} [\ell(\Psi; \boldsymbol{P}, z)]. \tag{9}$$

- Out-of-domain generalization: The testing queries follow $\mathcal{D}' \neq \mathcal{D}$, and the testing tasks follow $\mathcal{T}' \neq \mathcal{T}$. The generalization error is defined as

$$\mathop{\mathbb{E}}_{\boldsymbol{x}_{query} \sim \mathcal{D}', f \in \mathcal{T}'} [\ell(\Psi; \boldsymbol{P}, z)]. \tag{10}$$

Problem formulation

Model pruning:

- Let $\mathcal{S} \in [m]$ be the index set of $\boldsymbol{W}_O$ neurons.
- Pruning neurons in $\mathcal{S}$: removing corresponding rows of the trained $\boldsymbol{W}_O$.

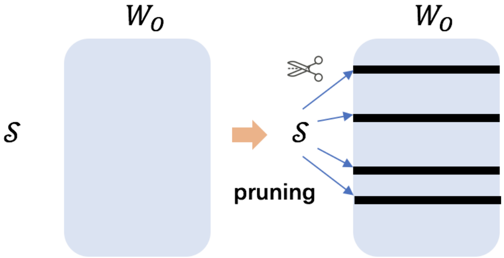


Figure 6: Pruning on W_O.

Formulating data and tasks

In-domain data and tasks:

- Given $\{\boldsymbol{\mu}_j\}_{j=1}^{M_1}$ as in-domain relevant (IDR) patterns, each in-domain data $\boldsymbol{x} = \boldsymbol{\mu}_j + \text{noise}$.
- Each task is defined based on one pair of $\boldsymbol{\mu}_a$ and $\boldsymbol{\mu}_b$. $f(\boldsymbol{x}) = +1$ (or -1) if the IDR pattern of $\boldsymbol{x}$ is $\boldsymbol{\mu}_a$ (or $\boldsymbol{\mu}_b$). $f(\boldsymbol{x})$ is a random label in other cases.

Out-of-domain data and tasks: Defined on out-of-domain relevant (ODR) patterns $\{\boldsymbol{\mu}'_j\}_{j=1}^{M'_1}$.

Prompt construction: For the task on $\boldsymbol{\mu}_a$ and $\boldsymbol{\mu}_b$, with a probability of $\alpha/2$, select examples of $\boldsymbol{\mu}_a$ and $\boldsymbol{\mu}_b$. α represents the fraction of task-relevant examples in the prompt. Replace α with α' if it is a testing task.

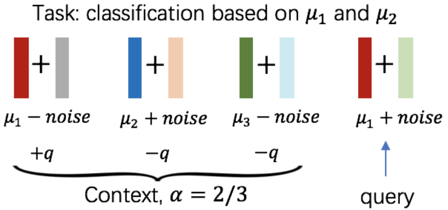


Figure 7: Example of prompt, $\alpha = 2/3$.

Main theoretical results

Theorem 1 (In-domain generalization)

For any $\epsilon > 0$, as long as

1. *the training tasks $\mathcal{T}_{tr}$ uniformly cover all the IDR patterns and labels with $|\mathcal{T}_{tr}|/|\mathcal{T}| \geq (M_1 - 1)^{-1/2}$, which means training a small fraction of the total tasks is sufficient,*
2. *the lengths of training and testing prompts $l_{tr} \geq \Omega(\alpha^{-1})$, $l_{ts} \geq \alpha'^{-1}$,*
3. *the number of iterations $T = \Theta(\alpha^{-2/3})$,*

and the batch size $B \geq \Omega(\max\{\epsilon^{-2}, M_1\})$, then with a high probability, the in-domain generalization error of the returned model is less than $\mathcal{O}(\epsilon)$.

ICL mechanism by the trained transformer

Proposition 1

- $\boldsymbol{W}_Q^{(T)}$ and $\boldsymbol{W}_K^{(T)}$ *mainly project context inputs to the IDR or ODR pattern.*
- *After training, attention weights become concentrated on contexts that share the same IDR/ODR pattern as the query. (induction head)*

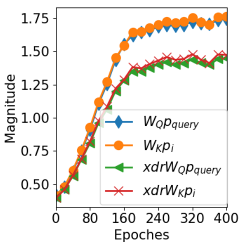


Figure 8: *The magnitude of the trained attention layer. xdr: IDR or ODR pattern of* p_{query}.

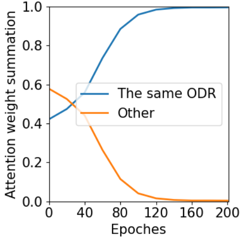


Figure 9: *The attention weight summation*

ICL mechanism by the trained transformer

Proposition 2

- *The feature embedding of rows of $\boldsymbol{W}_O^{(T)}\boldsymbol{W}_V^{(T)}$ approximate $\bar{\mu}$, i.e., the average of IDR patterns.*
- *The label embedding of rows $\boldsymbol{W}_O^{(T)}\boldsymbol{W}_V^{(T)}$ approximate $\boldsymbol{q}$ for positive neurons and $-\boldsymbol{q}$ for negative neurons.*

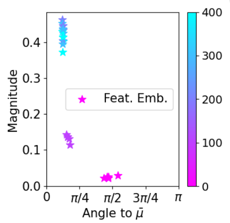


Figure 10: ***The feature embedding of $W_O W_V$. bar: iteration***

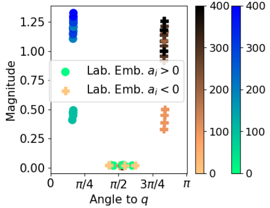


Figure 11: ***The label embedding of $W_O W_V$. bars: iterations***

Main theoretical results

Consider each ODR pattern as a linear combination of IDR patterns. Denote S_1 as the summation of the linear coefficients.

Theorem 2 (Out-of-domain generalization)

Suppose that the conditions (1) to (3) in Theorem 1 hold. If a constant order of $S_1 \geq 1$ and $l_{ts} \geq \alpha'^{-1}$, then with a high probability, the out-of-domain generalization error of the returned model is less than $\mathcal{O}(\epsilon)$.

Main theoretical results

Theorem 3 (Model pruning)

- *There exists a constant fraction of MLP-layer neurons of* $\boldsymbol{W}_O$ *with large weights, while the remaining have small weights.*
- *Pruning all neurons with small weights leads to a generalization error* $\mathcal{O}(\epsilon + M_2^{-1/2})$*, which is almost the same as without pruning.*
- *Pruning an* R *fraction of neurons with large weights results in a generalization error greater than* $\Omega(R)$*.*

Verifying the sufficient conditions for out-of-domain generalization.

- $S_1 \geq 1$ is needed for a desired out-of-domain generalization.
- The required length of testing prompts decreases as α' increases.

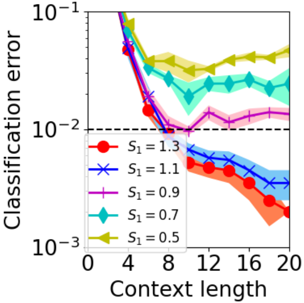


Figure 12: Out-of-domain ICL classification error on GPT-2 with different S_1

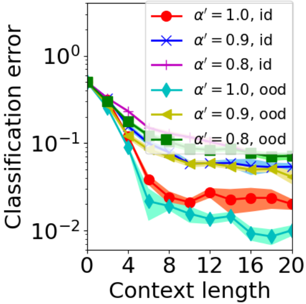


Figure 13: Out-of-domain ICL classification error on GPT-2 with different α'

Magnitude-based model pruning for out-of-domain ICL inference.

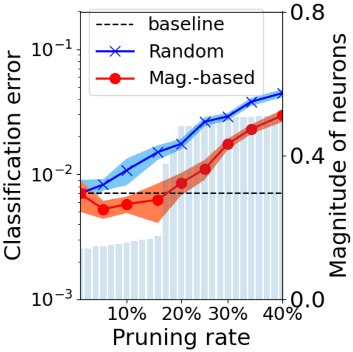


Figure 14: *Out-of-domain classification error with model pruning of the trained W_O and the magnitude of W_O neurons.*

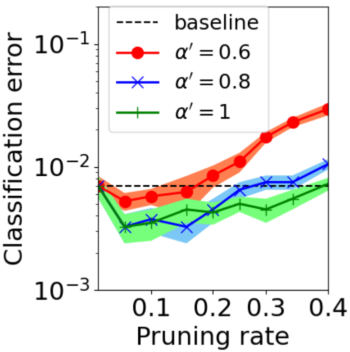


Figure 15: *Out-of-domain classification error with different α'*

Summary

- This work provides theoretical analyses of the training dynamics of Transformers with nonlinear attention and nonlinear MLP, and the resulting ICL capability for new tasks with possible data shift.
- This work also provides a theoretical justification for magnitude-based pruning to reduce inference costs while maintaining the ICL capability.
- This work provably characterizes the mechanism of ICL implemented by a single-head, one-layer Transformer.

Further exploration in LLM reasoning ability

Chain-of-Thought (COT)

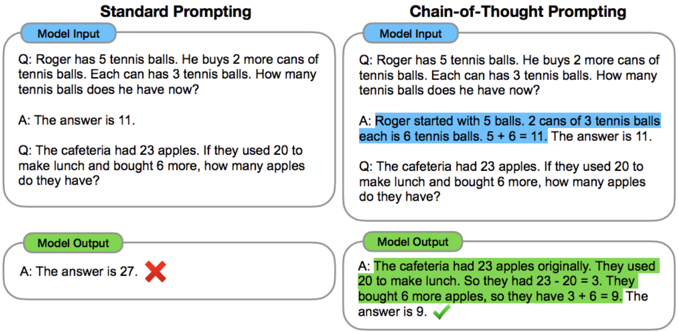


Figure 16: Few-shot COT [Wei et al.22]

Relationship with ICL: prompting multiple intermediate steps of reasoning.

Further exploration in LLM reasoning ability

Existing works focus on the expressive power of Transformer in implementing COT.

- [Li et el.23]: COT=Filtering+ICL.
- [Feng et al.23, Li et al.24]: Transformers can be constructed to solve many reasoning problems via COT.
- [Yang et al.24]: Linear Transformers can be more efficient than softmax Transformers in some dynamic programming tasks.

Problems to solve in our recent work[3]:

- How can a Transformer be trained to perform COT?
- When is COT better than ICL?
- Generalization with Data/Task distribution shift.

[3] https://arxiv.org/pdf/2410.02167

Further exploration in LLM reasoning ability

Problem formulation

Consider training on K-steps reasoning tasks $f = f_K \circ f_{K-1} \circ \cdots \circ f_2 \circ f_1$.

$\boldsymbol{P} = (\boldsymbol{E}_1, \boldsymbol{E}_2, \cdots, \boldsymbol{E}_{l_{tr}}, \boldsymbol{Q}_k)$ as the training prompt, where $\boldsymbol{E}_i = \begin{pmatrix} \boldsymbol{x}_i & \boldsymbol{y}_{i,1} & \cdots & \boldsymbol{y}_{i,K-1} \\ \boldsymbol{y}_{i,1} & \boldsymbol{y}_{i,2} & \cdots & \boldsymbol{y}_{i,K} \end{pmatrix}$ is the i-th context example, $\boldsymbol{Q}_k = \begin{pmatrix} \boldsymbol{z}_0 & \boldsymbol{z}_1 & \cdots & \boldsymbol{z}_{k-2} & \boldsymbol{z}_{k-1} \\ \boldsymbol{z}_1 & \boldsymbol{z}_2 & \cdots & \boldsymbol{z}_{k-1} & 0 \end{pmatrix}$ is the first k steps of the reasoning query for any k in $[K]$. The label for prediction is $\boldsymbol{z}_k$. Denote each column of $\boldsymbol{P}$ as $\boldsymbol{p}_i$. Add the positional encoding $\boldsymbol{c}_i$ (periodic) to each $\boldsymbol{p}_i$ to obtain $\tilde{\boldsymbol{p}}_i = \boldsymbol{p}_i + \boldsymbol{c}_{(i \mod K)}$.

Learning model:

$$f(\Psi; \boldsymbol{P}) = \sum_{i=1}^{\text{len}(P)-1} \boldsymbol{W}_V \tilde{\boldsymbol{p}}_i \text{softmax}((\boldsymbol{W}_K \tilde{\boldsymbol{p}}_i)^\top \boldsymbol{W}_Q \tilde{\boldsymbol{p}}_{query}) \tag{11}$$

Given training set $\{\boldsymbol{P}^n, z^n\}_{n=1}^N$. The loss is squared loss.

Further exploration in LLM reasoning ability

Problem formulation

The testing prompt $\boldsymbol{P} = (\boldsymbol{E}_1, \boldsymbol{E}_2, \cdots, \boldsymbol{E}_{l_{ts}}, \boldsymbol{p}_{query})$, where $\boldsymbol{p}_{query} = \begin{pmatrix} \boldsymbol{x}_{query} \\ 0 \end{pmatrix}$.

CoT inference: Feed the current prompt to the model to generate the most probable output $\boldsymbol{v}$ (greedy decoding), and then we put $\boldsymbol{v}$ at the end of $\boldsymbol{P}$ to form the new prompt.

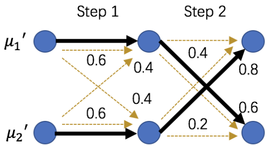
CoT Generalization error: the average error in each inference step $\mathbb{E}[\frac{1}{K}\sum_{k=1}^{K} \mathbb{1}[\boldsymbol{z}_k \neq \boldsymbol{v}_k]]$,

ICL inference: $\boldsymbol{E}_i = \begin{pmatrix} \boldsymbol{x}_i & 0 & \cdots & 0 \\ \boldsymbol{y}_{i,K} & 0 & \cdots & 0 \end{pmatrix}$ is the i-th context example. The ICL generalization error: $\mathbb{E}[\mathbb{1}[\boldsymbol{z}_k \neq \boldsymbol{v}]]$.

Further exploration in LLM reasoning ability

Data modeling

The training tasks are the transition between M training-relevant (TRR) patterns $\boldsymbol{\mu}_i$. The testing tasks are the transition between M' testing-relevant (TSR) patterns $\boldsymbol{\mu}'_i$.



Testing examples contain erroneous steps, and transition matrices characterize the transition. Examples: correct paths are $\boldsymbol{\mu}'_1 \to \boldsymbol{\mu}'_1 \to \boldsymbol{\mu}'_2$, $\boldsymbol{\mu}'_2 \to \boldsymbol{\mu}'_2 \to \boldsymbol{\mu}'_1$. Step-wise transition matrices: $\boldsymbol{A}_1^f = \begin{pmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{pmatrix}$, $\boldsymbol{A}_2^f = \begin{pmatrix} 0.4 & 0.6 \\ 0.8 & 0.2 \end{pmatrix}$. K-steps transition matrix: $\boldsymbol{B}^f = \begin{pmatrix} 0.56 & 0.44 \\ 0.64 & 0.36 \end{pmatrix}$. τ^f: min-max trajectory transition probability, $\tau^f = 0.36$. τ_o^f: min-max input-label transition probability, $\tau_o^f = 0.56$.

Further exploration in LLM reasoning ability

Theoretical Results

Define α and α' as the fraction of context examples with input sharing the same TRR and TSR pattern as the query input, respectively.

Theorem 4

For any $\epsilon > 0$, as long as

1. *the training tasks and samples are selected such that every TRR pattern is equally likely in every inference step and in each training batch,*
2. *the length of training prompts $l_{tr} \geq \Omega(\alpha^{-1})$*
3. *and the number of iterations $T = \Theta(\alpha^{-2}K^3 + MK(\alpha^{-1} + \epsilon^{-1}))$,*

and the batch size $B \geq \Omega(\epsilon^{-2})$, then with a high probability, the loss of the returned model is less than $\mathcal{O}(\epsilon)$.

Theoretical Results

Theorem 5 (CoT generalization)

As long as

1. *each TSR pattern* $\boldsymbol{\mu}'_i$ *is a linear combination of all the TRR pattern* $\boldsymbol{\mu}_i$,
2. *the length of testing prompts* $l_{ts} \geq \Omega((\alpha' \tau^f)^{-2})$

then with a high probability, we have the CoT generalization error $= 0$.

A more informative prompt (larger α') and more accurate inference examples (larger τ^f) can reduce the required testing prompt length.

Further exploration in LLM reasoning ability

Theoretical Results

Comparison with ICL:

We first propose Condition 1: the correct final output is the most probable output by $\boldsymbol{B}^f$. The previous condition does not satisfy this condition.

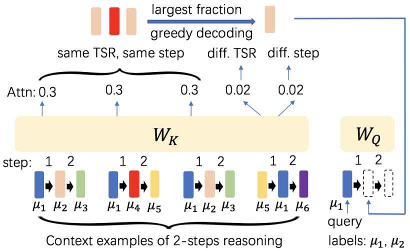
Theorem 6 (ICL generalization)

1. *If condition 1 does not hold, then the ICL generalization error* $\geq \Omega(1)$.
2. *If condition 1 holds, and* $l_{ts} \geq \Omega((\alpha' \tau_o^f)^{-2})$, *we have the ICL generalization error* $= 0$.

Because Condition 1 is not required for CoT generalization, CoT performs better than ICL if Condition 1 fails.

Further exploration in LLM reasoning ability

CoT Mechanism



1. When conducting the k-th step reasoning of the query, the trained model assigns dominant attention weights on the prompt columns that are also the k-th step and share the same TSR pattern as the query.
2. Then, the fraction of the correct TSR pattern is the largest in the output of each step to generate the accurate output by greedy decoding.

Experiments

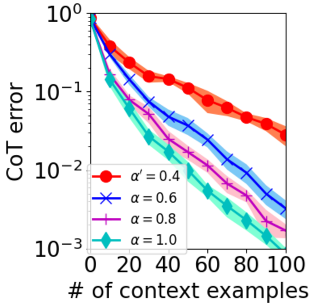


Figure 17: ***CoT testing error with different α'***

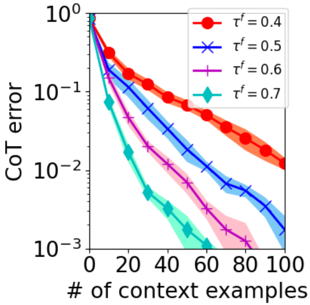


Figure 18: ***CoT testing error with different τ***

More testing examples are needed when α' or τ^f is small.

Experiments

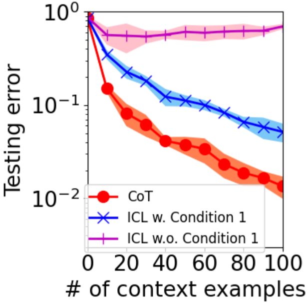


Figure 19: Comparison between CoT and ICL w./w.o. Condition 1

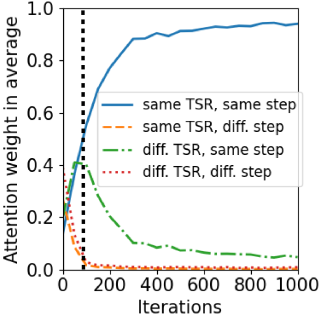


Figure 20: Mechanism of Transformers for CoT

Summary

- This work provides the training dynamics analysis of nonlinear Transformer towards CoT generalization.
- This work also characterizes the requirements for a guaranteed CoT generalization with a provable mechanism.
- This work theoretically studies when CoT is better than ICL.

Future Directions

Some interesting high-level insights:

The low dimensionality of language data leads to the following results of Transformers.

1. Induction Head: Concentrated attention+copying in in/Out-of-domain inference.
2. Sparsity: Neurons only learn a few patterns.

The reason why CoT works is CoT can do "matching and copying" rather than learning any "logic" from data.

Future directions:

- What is the mechanism of ICL/CoT in more general generation tasks?
- Can CoT learn a more complicated reasoning structure provably?
- Does CoT really make inferences by copying known tokens instead of from any logic that CoT learns?

Thank you!

Q & A

Wayne Xin Zhao, Kun Zhou*, Junyi Li*, Tianyi Tang, Xiaolei Wang, et al.
A Survey of Large Language Models
https://arxiv.org/pdf/2303.18223.pdf

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, et al.
Language Models are Few-Shot Learners
OpenAI.

Shivam Garg, Dimitris Tsipras, Percy Liang, Gregory Valiant
What Can Transformers Learn In-Context? A Case Study of Simple Function Classes.
In *Advances in Neural Information Processing Systems 2022.*

Ekin Akyurek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, Denny Zhou
What learning algorithm is in-context learning? Investigations with linear models
In *International conference on Learning Representations 2023.*

Ruiqi Zhang, Spencer Frei, Peter L. Bartlett
Trained transformers learn linear models in-context
In *Journal of Machine Learning Research*

Jingfeng Wu, Difan Zou, Zixiang Chen, Vladimir Braverman, Quanquan Gu, Peter L. Bartlett
How many pretraining tasks are needed for in-context learning of linear regression?
In *International conference on Learning Representations 2024.*

Yu Huang, Yuan Cheng, Yingbin Liang
In-context convergence of transformers.
In *International conference on Machine Learning 2024.*

Johannes von Oswald, Eyvind Niklasson, Ettore Randazzo, Joao Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, Max Vladymyrov
Transformers Learn In-Context by Gradient Descent.
In *International conference on Machine Learning 2023.*

Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, Suvrit Sra
Transformers learn to implement preconditioned gradient descent for in-context learning.
In *Neurips 2023.*

Xiang Cheng, Yuxin Chen, Suvrit Sra

Transformers Implement Functional Gradient Descent to Learn Non-Linear Functions In Context.
In *International conference on Machine Learning 2024.*

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph et al.
In-context Learning and Induction Heads.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter et al.
Chain-of-Thought Prompting Elicits Reasoning in Large Language Models
In *Neurips 2022.*

Yingcong Li, Kartik Sreenivasan, Angeliki Giannou, Dimitris Papailiopoulos, Samet Oymak
Dissecting Chain-of-Thought: Compositionality through In-Context Filtering and Learning
In *Neurips 2023.*

Zhiyuan Li, Hong Liu, Denny Zhou, Tengyu Ma
Chain of Thought Empowers Transformers to Solve Inherently Serial Problems
In *International conference on Learning Representations 2024.*

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, Liwei Wang

Towards Revealing the Mystery behind Chain of Thought: A Theoretical Perspective
In *Neurips 2023.*

Kai Yang, Jan Ackermann, Zhenyu He, Guhao Feng, Bohang Zhang, Yunzhen Feng, Qiwei Ye, Di He, and Liwei Wang.
Do efficient transformers really save computation?
https://arxiv.org/pdf/2402.13934.pdf

Backup

Proof idea of Theorem 1

Analytical Framework: Feature learning

1. Assuming a mapping from different patterns to different labels.
2. Characterize the gradient updates, which will be proven to be significant in the directions of patterns that determine the labels.
3. The accumulated gradient updates will lead to different types of trained neurons, which have different impacts on learning.

High-level idea to prove Theorem 1

1. Characterize the gradient updates of $\boldsymbol{W}_Q$, $\boldsymbol{W}_K$, $\boldsymbol{W}_V$, and $\boldsymbol{W}_O$ in terms of IDR patterns.
2. We show the model makes attention weights converge to 1 between the same IDR patterns and the MLP layer makes predictions based on the label embedding.

Proof idea of Theorem 1

Self-attention layer

$$
\begin{aligned}
&\eta \frac{1}{B} \sum_{n \in \mathcal{B}_b} \frac{\partial \ell(\tilde{\boldsymbol{P}}^n, z^n; \Psi)}{\partial \boldsymbol{W}_Q} \\
&= \eta \frac{1}{B} \sum_{n \in \mathcal{B}_b} (-z^n) \sum_{i=1}^{m} a_i \mathbb{1}[\boldsymbol{W}_{O_{(i,\cdot)}} \sum_{s=1}^{l+1} (\boldsymbol{W}_V \boldsymbol{p}_s^n) \text{softmax}(\boldsymbol{p}_s^{n\top} \boldsymbol{W}_K^\top \boldsymbol{W}_Q \boldsymbol{p}_{query}^n) \geq 0] \\
&\cdot \Big(\boldsymbol{W}_{O_{(i,\cdot)}} \sum_{s=1}^{l+1} (\boldsymbol{W}_V \boldsymbol{p}_s^n) \text{softmax}(\boldsymbol{p}_s^{n\top} \boldsymbol{W}_K^\top \boldsymbol{W}_Q \boldsymbol{p}_{query}^n) \\
&\cdot (\boldsymbol{W}_K \boldsymbol{p}_s^n - \sum_{r=1}^{l+1} \text{softmax}(\boldsymbol{p}_r^{n\top} \boldsymbol{W}_K^\top \boldsymbol{W}_Q \boldsymbol{p}_{query}^n) \boldsymbol{W}_K \boldsymbol{p}_r^n) \boldsymbol{p}_{query}^\top \Big).
\end{aligned}
$$

1. Consider $z^n = 1$, $a_i > 0$, $\mathbb{1}[\cdot] = 1$ ("lucky" neurons, will be introduced later), which gives a positive gradient gain of the last two rows.
2. If the attention weights between $\boldsymbol{p}_s^n$ and $\boldsymbol{p}_{query}^n$ is large with $\boldsymbol{p}_s^n$ sharing the same IDR pattern as $\boldsymbol{p}_{query}^n$, then $-grad(\boldsymbol{W}_Q) \cdot \boldsymbol{p}_{query} \propto \boldsymbol{p}_{query}$ approximately as desired.

Proof idea of Theorem 1

Self-attention layer

$$
\begin{aligned}
&\eta\frac{1}{B}\sum_{n\in\mathcal{B}_b}\frac{\partial\ell(\tilde{\boldsymbol{P}}^n,z^n,\Psi)}{\partial\boldsymbol{W}_K}\\
=&\eta\frac{1}{B}\sum_{n\in\mathcal{B}_b}(-z^n)\sum_{i=1}^m a_i\mathbb{1}[\boldsymbol{W}_{O_{(i,\cdot)}}\sum_{s=1}^{l+1}(\boldsymbol{W}_V\boldsymbol{p}_s^n)\cdot\text{softmax}({\boldsymbol{p}_s^n}^\top\boldsymbol{W}_K^\top\boldsymbol{W}_Q\boldsymbol{p}_{query}^n)\geq 0]\\
&\cdot\Big(\boldsymbol{W}_{O_{(i,\cdot)}}\sum_{s=1}^{l+1}(\boldsymbol{W}_V\boldsymbol{p}_s^n)\text{softmax}({\boldsymbol{p}_s^n}^\top\boldsymbol{W}_K^\top\boldsymbol{W}_Q\boldsymbol{p}_{query})\boldsymbol{W}_Q^\top\boldsymbol{p}_{query}^n\\
&\cdot(\boldsymbol{p}_s^n-\sum_{r=1}^{l+1}\text{softmax}({\boldsymbol{p}_r^n}^\top\boldsymbol{W}_K^\top\boldsymbol{W}_Q\boldsymbol{p}_{query}^n)\boldsymbol{p}_r^n)^\top\Big).
\end{aligned}
$$

1. If the attention weights between $\boldsymbol{p}_s^n$ and $\boldsymbol{p}_r^n$ is large with $\boldsymbol{p}_s^n$ sharing the same IDR pattern as $\boldsymbol{p}_r^n$, then $-grad(\boldsymbol{W}_K)\cdot\boldsymbol{p}_r\propto\boldsymbol{p}_r$ approximately as desired.
2. Combining the result of $\boldsymbol{W}_Q$, this will in turn enlarge the attention weights between $\boldsymbol{p}_{query}^n$ and $\boldsymbol{p}_s^n$ of the same IDR pattern. An induction can prove this process.

Proof idea of Theorem 1

What does the attention layer imply from the gradient update?

The weighted summation of $\boldsymbol{p}_s^n$ with attention as coefficients has the following property.

1. The feature embedding part will be close to the IDR pattern of $\boldsymbol{p}_{query}^n$, while the IDI pattern is filtered out.
2. The label embedding part will be close to the label of $\boldsymbol{p}_s^n$ that shares the same IDR pattern as $\boldsymbol{p}_{query}^n$. This implies that it will be great if $\boldsymbol{W}_O\boldsymbol{W}_V$ makes predictions only based on the label embedding. In fact, it is true!

Proof idea of Theorem 1

MLP layer ($\boldsymbol{W}_V$ included. It is highly correlated with $\boldsymbol{W}_O$.)

$$
\begin{aligned}
&\eta \frac{1}{B} \sum_{n \in \mathcal{B}_b} \frac{\partial \ell(\tilde{\boldsymbol{P}}^n, z^n; \Psi)}{\partial \boldsymbol{W}_V} \\
&= \eta \frac{1}{B} \sum_{n \in \mathcal{B}_b} (-z^n) \sum_{i=1}^{m} a_i \mathbb{1}[\boldsymbol{W}_{O_{(i,\cdot)}} \sum_{s=1}^{l+1} (\boldsymbol{W}_V \boldsymbol{p}_s^n) \text{softmax}({\boldsymbol{p}_s^n}^\top \boldsymbol{W}_K^\top \boldsymbol{W}_Q \boldsymbol{p}_{query}^n) \geq 0] \\
&\cdot \boldsymbol{W}_{O_{(i,\cdot)}}^\top \sum_{s=1}^{l+1} \text{softmax}({\boldsymbol{p}_s^n}^\top \boldsymbol{W}_K^\top \boldsymbol{W}_Q \boldsymbol{p}_{query}^n) {\boldsymbol{p}_s^n}^\top.
\end{aligned}
$$

1. The projection of *Grad*($\boldsymbol{W}_V$) onto different IDR patterns replies on $\boldsymbol{W}_{O_{(i,\cdot)}}$ for different i.

Proof idea of Theorem 1

MLP layer

How to formulate different $\boldsymbol{W}_O$ neurons?

We characterize "lucky neurons", i.e., some rows of $\boldsymbol{W}_O$, which are initialized such that at the beginning of the training, the indicator function $\mathbb{1}[\boldsymbol{W}_O \sum_s (\boldsymbol{W}_V \boldsymbol{p}_s) softmax(\boldsymbol{p}_s^\top \boldsymbol{W}_K^\top \boldsymbol{W}_Q \boldsymbol{p}_{query}) \geq 0]$ is activated. See definition D.8.

Properties of lucky neurons

1. The fraction of lucky neurons $\geq \Omega(1)$.
2. During the training, the label embedding becomes approximately in the direction of $\boldsymbol{q}$ or $-\boldsymbol{q}$ for $a_i > 0$ or $a_i < 0$, respectively.
3. The feature embedding gradually becomes the average of IDR patterns along the training.

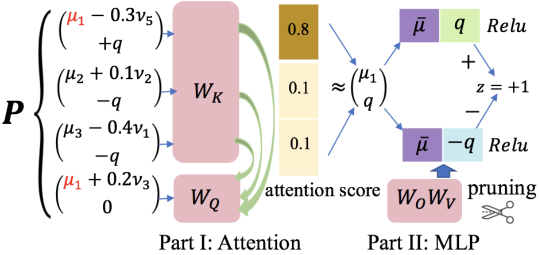
Proof idea of Theorem 1

MLP layer

$$
\begin{aligned}
&\eta\frac{1}{B}\sum_{n\in\mathcal{B}_b}\frac{\partial \ell(\tilde{\boldsymbol{P}}^n, z^n;\Psi)}{\partial \boldsymbol{W}_{O_{(i,\cdot)}}}\\
=&\eta\frac{1}{B}\sum_{n\in\mathcal{B}_b}(-z^n)a_i\mathbb{1}[\boldsymbol{W}_{O_{(i,\cdot)}}\sum_{s=1}^{l+1}(\boldsymbol{W}_V\boldsymbol{p}_s^n)\text{softmax}(\boldsymbol{p}_s^{n\top}\boldsymbol{W}_K^\top\boldsymbol{W}_Q\boldsymbol{p}_{query}^n)\geq 0]\\
&\cdot\sum_{s=1}^{l+1}(\boldsymbol{W}_V\boldsymbol{p}_s^n)\text{softmax}(\boldsymbol{p}_s^{n\top}\boldsymbol{W}_K^\top\boldsymbol{W}_Q\boldsymbol{p}_{query}^n).
\end{aligned}
$$

1. We can use an induction to prove the gradient update by combining the changes of $\boldsymbol{W}_V$.
2. Lucky neurons of $+\boldsymbol{q}$ will grow approximately in the direction of $\boldsymbol{W}_V\boldsymbol{p}_s$ of $+\boldsymbol{q}$, which further enhances such a direction. The same for lucky neurons of $-\boldsymbol{q}$.
3. Unlucky neurons has small weights due to unstable a_i and $\mathbb{1}[\cdot]$.

To sum up



1. Attention weights between the same IDR pattern, i.e., $\boldsymbol{\mu}_1 + 0.2\boldsymbol{v}_3$ and $\boldsymbol{\mu}_1 - 0.3\boldsymbol{v}_5$, become dominant, resulting in a weighted summation close to $(\boldsymbol{\mu}_1^\top, \boldsymbol{q}^\top)^\top$.
2. Lucky neurons are proved to be either $(\bar{\boldsymbol{\mu}}^\top, \boldsymbol{q}^\top)^\top$ or $(\bar{\boldsymbol{\mu}}^\top, -\boldsymbol{q}^\top)^\top$. This leads to a correct prediction given $(\boldsymbol{\mu}_1^\top, \boldsymbol{q}^\top)^\top$ as the input.

Proof idea of Theorem 2

1. Each ODR pattern as a linear combination of IDR patterns: ensure Proposition 1 still holds for ODR patterns.
2. $S_1 \geq 1$ allows the lucky neurons still activated: Approximately,

$$\begin{aligned} \boldsymbol{W}_O^{(T)}\boldsymbol{W}_V^{(T)}(\boldsymbol{\mu}_1'^\top, \boldsymbol{q}^\top) \approx & \bar{\boldsymbol{\mu}}^\top \boldsymbol{\mu}_1' + \boldsymbol{q}^\top \boldsymbol{q} \\ = & \bar{\boldsymbol{\mu}}^\top \sum_{i=1}^{M} c_i \boldsymbol{\mu}_i + \boldsymbol{q}^\top \boldsymbol{q} \\ = & \sum_{i=1}^{M} c_i \bar{\boldsymbol{\mu}}^\top \boldsymbol{\mu}_i + \boldsymbol{q}^\top \boldsymbol{q} \\ \geq & \bar{\boldsymbol{\mu}}^\top \boldsymbol{\mu}_1 + \boldsymbol{q}^\top \boldsymbol{q} \end{aligned} \tag{12}$$

ICL mechanism by the trained transformer

Results of multi-layer Transformers (3-layer).

- Each attention layer selects contexts with the same IDR pattern as the query.

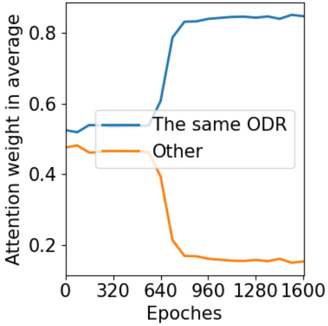


Figure 21: ***Layer 1 self-attention***

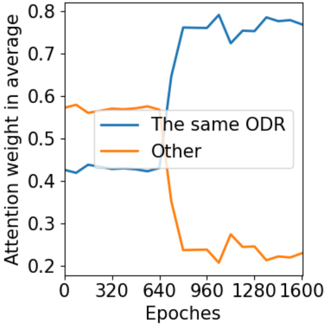


Figure 22: ***Layer 2 self-attention***

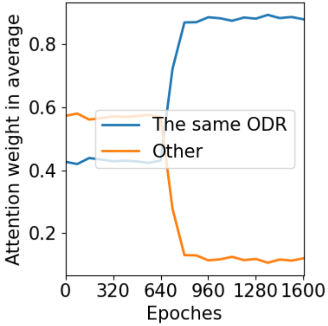


Figure 23: ***Layer 3 self-attention***

ICL mechanism by the trained transformer

Results of multi-layer Transformers (3-layer).

- The magnitude of the majority of neurons increases along the training.
- The angle changes still hold for one of the layers.

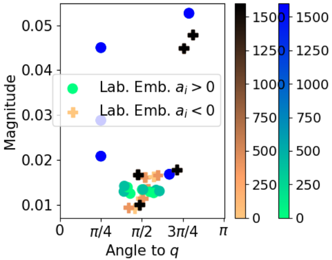


Figure 24: Layer 1 self-attention

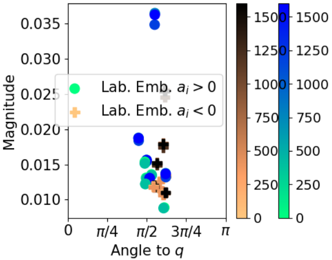


Figure 25: Layer 2 self-attention

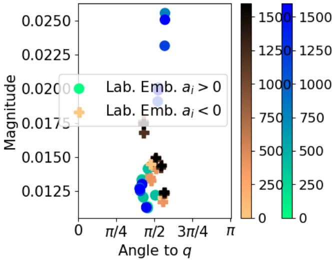


Figure 26: Layer 3 self-attention

Comparing ICL on a one-layer Transformer with other machine learning algorithms.

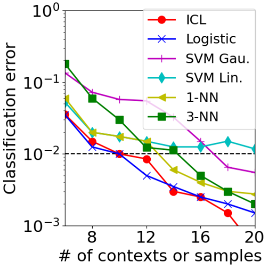


Figure 27: *Binary classification performance of using different algorithms,* $\alpha' = 0.8$

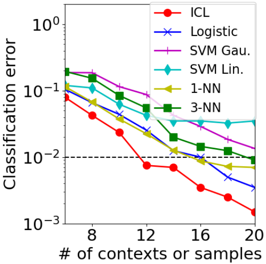


Figure 28: *Binary classification performance of using different algorithms,* $\alpha' = 0.6$

- Logistic: logistic regression; SVM Gau.: SVM with Gaussian kernel; SVM Lin.: SVM with linear kernel; 1-NN: 1-nearest neighbor; 3-NN: 3-nearest neighbor.