In-Context Learning State Vector with Inner and Momentum Optimization

Dongfang Li, Zhenyu Liu, Xinshuo Hu,
Zetian Sun, Baotian Hu*, Min Zhang

Harbin Institute of Technology (Shenzhen),

Email: liuzhenyuhit@gmail.com

Shenzhen, China

*Corresponding author

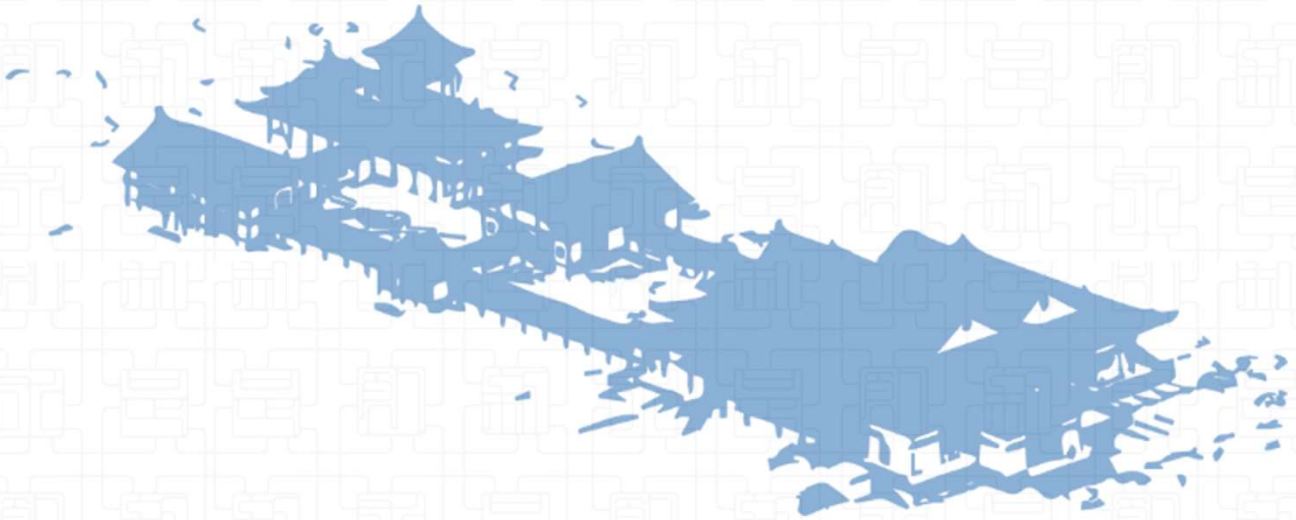
Presented at NeurIPS 2024

CONTENTS

1

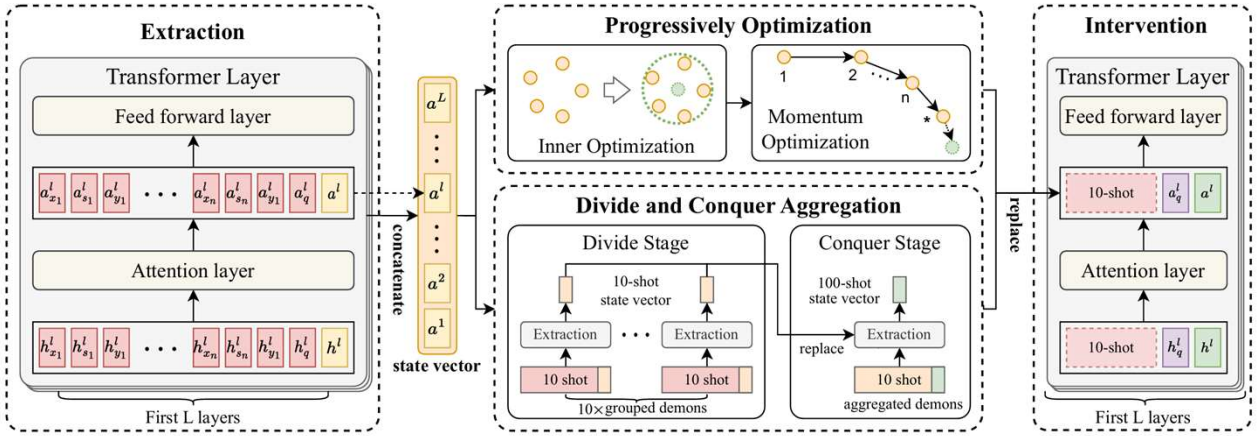
Preamble

- Overview
- Motivation



1.1 Overview

In this study, we explore the in-context learning compression vector and propose novel methods for its optimization and aggregation.

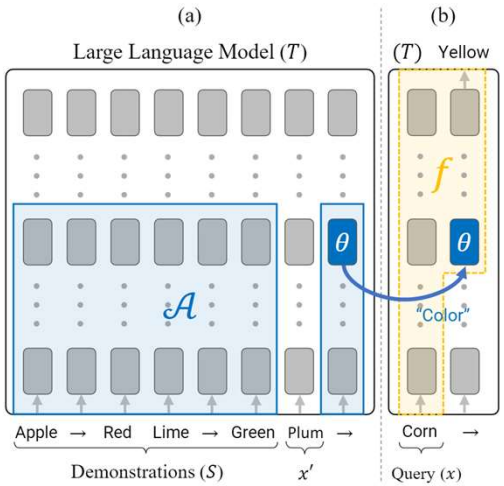


1.2 Motivation

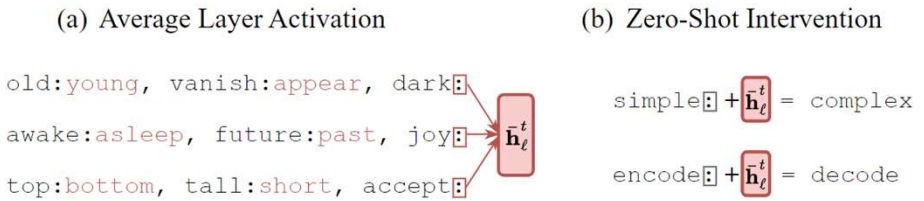
哈爾濱工業大學 HARBIN INSTITUTE OF TECHNOLOGY

In-context learning compression vector

- Recent work[1][2] shows that task information from ICL is stored in the early transformer layers
- The ICL vector enables the model to perform In-Context Learning without requiring explicit demonstrations.



[1] In-Context Learning Creates Task Vectors



[2] Function Vectors in Large Language Models

1.2 Motivation

哈爾濱工業大學 HARBIN INSTITUTE OF TECHNOLOGY

In-context learning compression vector

- Recent work shows that task information from ICL is stored in the early transformer layers
- The ICL vector enables the model to perform In-Context Learning without requiring explicit demonstrations.

However, the working mechanisms and optimization of these vectors are yet to be thoroughly explored.

1.2 Motivation

哈爾濱工業大學 HARBIN INSTITUTE OF TECHNOLOGY

In-context learning compression vector

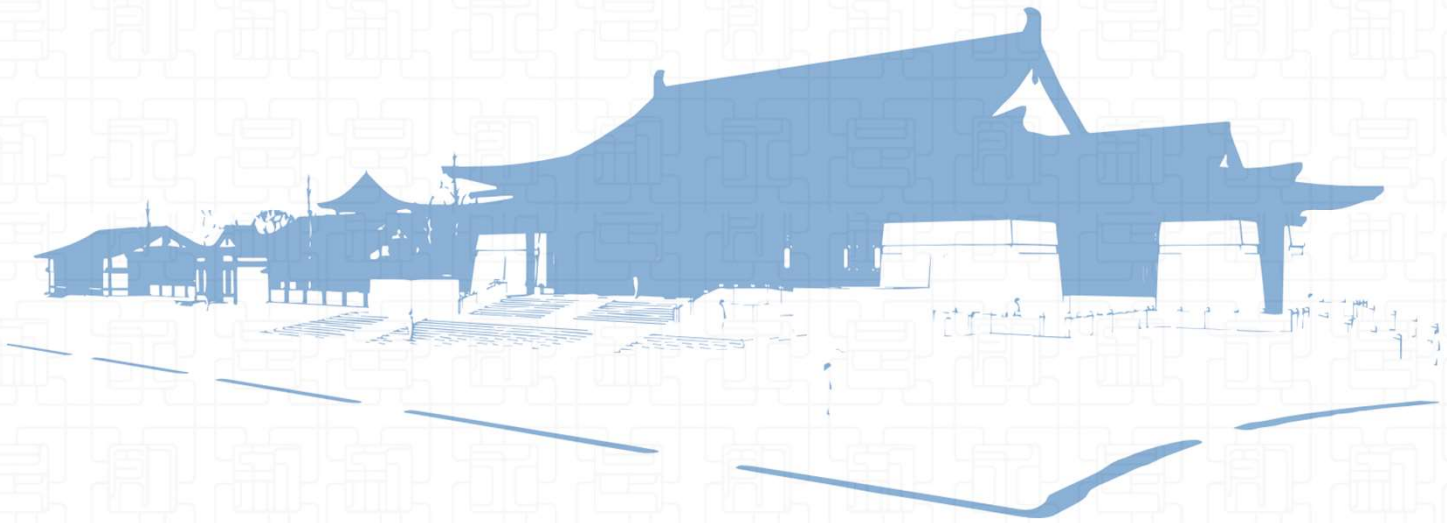
- Recent work shows that task information from ICL is stored in the early transformer layers
- The ICL vector enables the model to perform In-Context Learning without requiring explicit demonstrations.

1. Why do these ICL vectors work?

2. How can we improve the performance of these ICL vectors?

2
Methodology

- Formalization
- Method



2.1 Formalization

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

Classic template of ICL:

$$\mathcal{X}_1, \mathcal{S}, \mathcal{Y}_1, \mathcal{X}_2, \mathcal{S}, \mathcal{Y}_2, \cdots, \mathcal{X}_N, \mathcal{S}, \mathcal{Y}_N, \mathcal{X}_q, \mathcal{S}.$$

2.1 Formalization

哈爾濱工業大學 HARBIN INSTITUTE OF TECHNOLOGY

Classic template of ICL:

$$\mathcal{X}_1, \mathcal{S}, \mathcal{Y}_1, \mathcal{X}_2, \mathcal{S}, \mathcal{Y}_2, \cdots, \mathcal{X}_N, \mathcal{S}, \mathcal{Y}_N, \mathcal{X}_q, \mathcal{S}.$$

Output attention activation of the last separate token:

$$\mathbf{a}^l = W_V[X'; X] \operatorname{softmax}\left(\frac{(W_K[X'; X])^T \mathbf{q}}{\sqrt{d}}\right),$$

2.1 Formalization

哈爾濱工業大學 HARBIN INSTITUTE OF TECHNOLOGY

Classic template of ICL:

$$\mathcal{X}_1, \mathcal{S}, \mathcal{Y}_1, \mathcal{X}_2, \mathcal{S}, \mathcal{Y}_2, \cdots, \mathcal{X}_N, \mathcal{S}, \mathcal{Y}_N, \mathcal{X}_q, \mathcal{S}.$$

Output attention activation of the last separate token:

$$\mathbf{a}^l = W_V[X';X]\,\mathrm{softmax}\left(\frac{(W_K[X';X])^T\,\mathbf{q}}{\sqrt{d}}\right),$$

Omit the softmax operation and the scaling factor:

$$\begin{aligned}\mathbf{a}^l &\approx W_V[X';X]\,(W_K[X';X])^T\,\mathbf{q}\\ &= \left(W_V X\,(W_K X)^T + W_V X'\,(W_K X')^T\right)\mathbf{q}\\ &= \left(W_{\mathrm{ZSL}} + \sum_i ((W_V\mathbf{x}_i') \otimes (W_K\mathbf{x}_i'))\right)\mathbf{q}.\end{aligned}$$

2.1 Formalization

$$
\begin{aligned}
\mathbf{a}^l &\approx W_V[X';X]\left(W_K[X';X]\right)^T\mathbf{q} \\
&= \left(W_V X\left(W_K X\right)^T + W_V X'\left(W_K X'\right)^T\right)\mathbf{q} \\
&= \left(W_{\mathrm{ZSL}} + \sum_i \left(\left(W_V\mathbf{x}'_i\right)\otimes\left(W_K\mathbf{x}'_i\right)\right)\right)\mathbf{q}.
\end{aligned}
$$

back-propagated errors:

$$
\Delta W_{GD} = \sum_i \mathbf{e}_i \otimes \mathbf{x}'_i,
$$

$$
\mathbf{a}^l = \left(W_{\mathrm{ZSL}} + \sum_i \mathbf{e}_i \otimes W_K\mathbf{x}'_i\right)\mathbf{q} = \left(W_{\mathrm{ZSL}} + \Delta W_{GD}\right)\mathbf{q}.
$$

2.1 Formalization

$$\begin{aligned}\mathbf{a}^l &\approx W_V[X';X]\left(W_K[X';X]\right)^T\mathbf{q}\\ &= \left(W_V X\left(W_K X\right)^T + W_V X'\left(W_K X'\right)^T\right)\mathbf{q}\\ &= \left(W_{\text{ZSL}} + \sum_i\left(\left(W_V\mathbf{x}'_i\right)\otimes\left(W_K\mathbf{x}'_i\right)\right)\right)\mathbf{q}.\end{aligned}$$

back-propagated errors:

$$\Delta W_{GD} = \sum_i \mathbf{e}_i \otimes \mathbf{x}'_i,$$

$$\mathbf{a}^l = \left(W_{\text{ZSL}} + \sum_i \mathbf{e}_i \otimes W_K\mathbf{x}'_i\right)\mathbf{q} = \left(W_{\text{ZSL}} + \Delta W_{GD}\right)\mathbf{q}.$$

It can be inferred that the output activation can be regarded as parameters trained via gradient descent which utilizes the demonstrations as training instances.

2.1 Formalization

$$
\begin{aligned}
\mathbf{a}^l &\approx W_V[X';X]\left(W_K[X';X]\right)^T\mathbf{q} \\
&= \left(W_V X\left(W_K X\right)^T + W_V X'\left(W_K X'\right)^T\right)\mathbf{q} \\
&= \left(W_{\text{ZSL}} + \sum_i \left(\left(W_V \mathbf{x}'_i\right) \otimes \left(W_K \mathbf{x}'_i\right)\right)\right)\mathbf{q}.
\end{aligned}
$$

back-propagated errors:

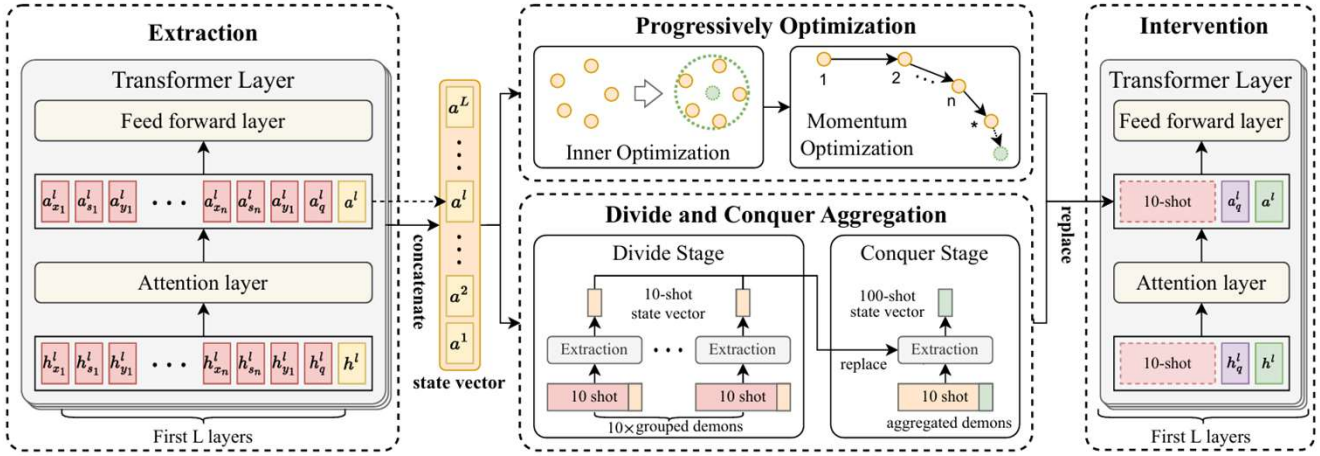
$$
\Delta W_{GD} = \sum_i \mathbf{e}_i \otimes \mathbf{x}'_i,
$$

$$
\mathbf{a}^l = \left(W_{\text{ZSL}} + \sum_i \mathbf{e}_i \otimes W_K \mathbf{x}'_i\right)\mathbf{q} = \left(W_{\text{ZSL}} + \Delta W_{GD}\right)\mathbf{q}.
$$

State vector:

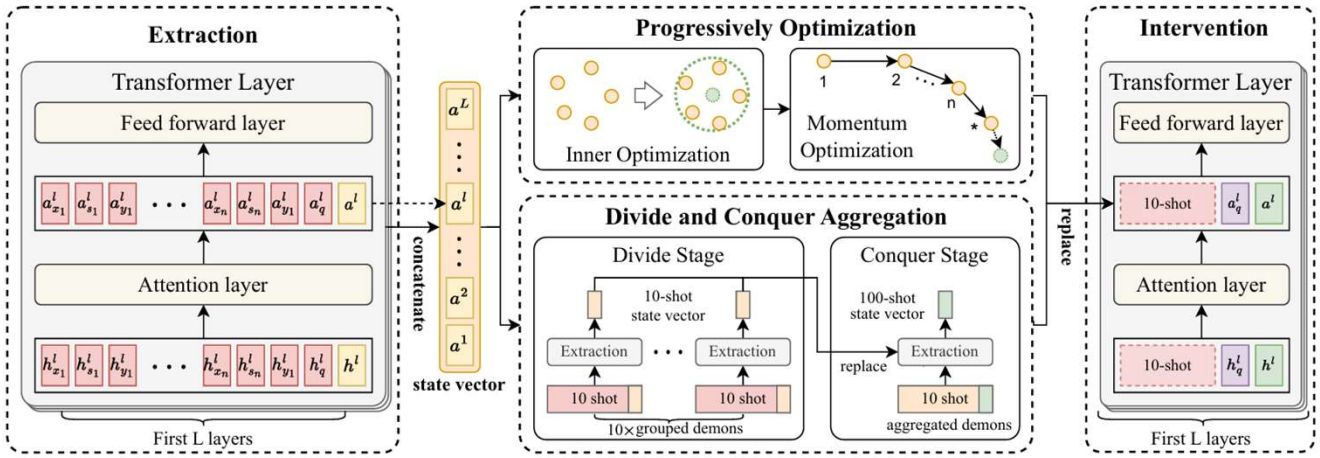
$$
\mathcal{V}_N^L = \Big\Vert_{l=1}^{L} \mathbf{a}^l,
$$

2.2 Method



- **Extraction** to obtain the state vector from an ICL feed-forward pass.
- **Intervention** to enable ICL tasks without any demonstrations
- **Progressive Optimization** to enhance the state vector's effectiveness
- **Divide and Conquer Aggregation** to derive state vectors from multiple examples

2.2 Method

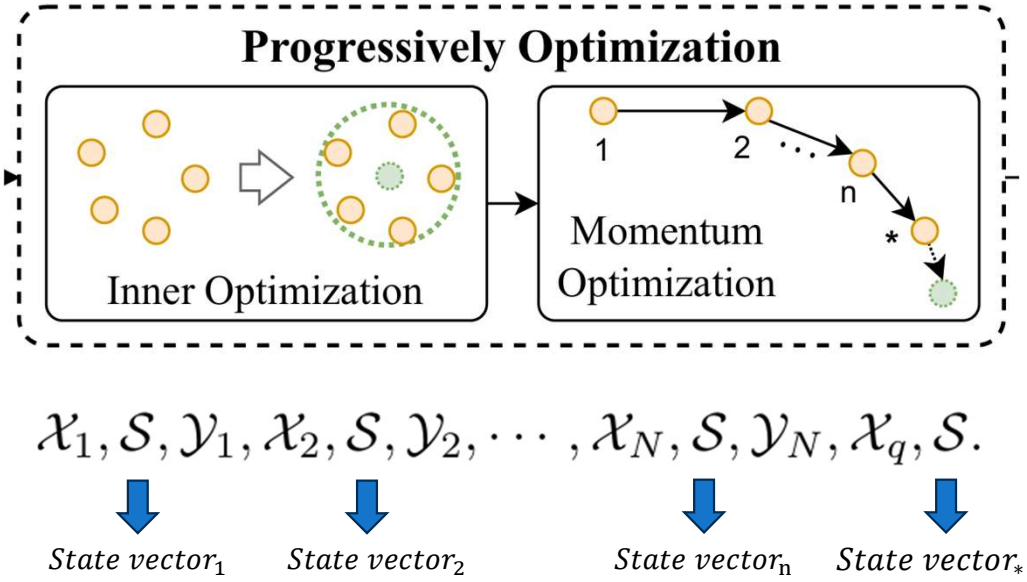


- **Extraction** to obtain the state vector from an ICL feed-forward pass.
- **Intervention** to enable ICL tasks without any demonstrations
- **Progressive Optimization** to enhance the state vector's effectiveness
- **Divide and Conquer Aggregation** to obtain state vectors from multiple examples

2.2 Method

哈爾濱工業大學 HARBIN INSTITUTE OF TECHNOLOGY

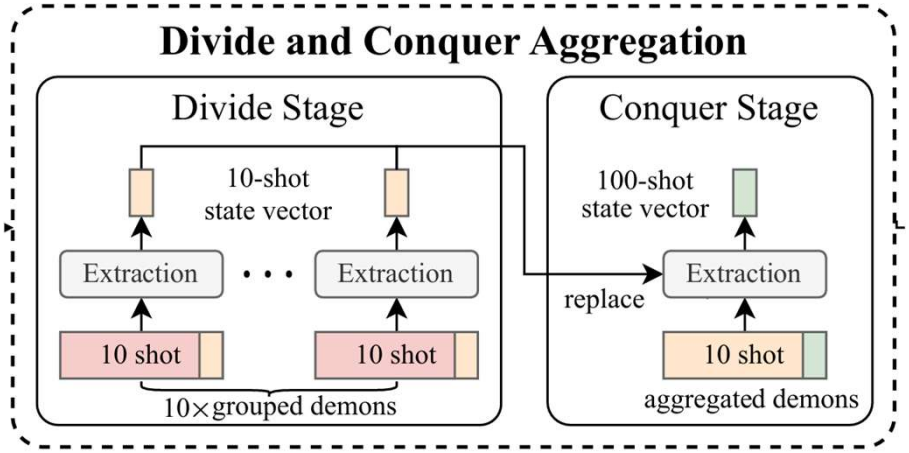
- Inspired by model soup and momentum-based gradient optimization, we introduce two methods for state vector refinement:
  - **Inner Optimization** averages state vectors across separator tokens to improve robustness within a single pass
  - **Momentum Optimization** iteratively updates the state vector by capturing changes across separator tokens, simulating gradient-based enhancement.



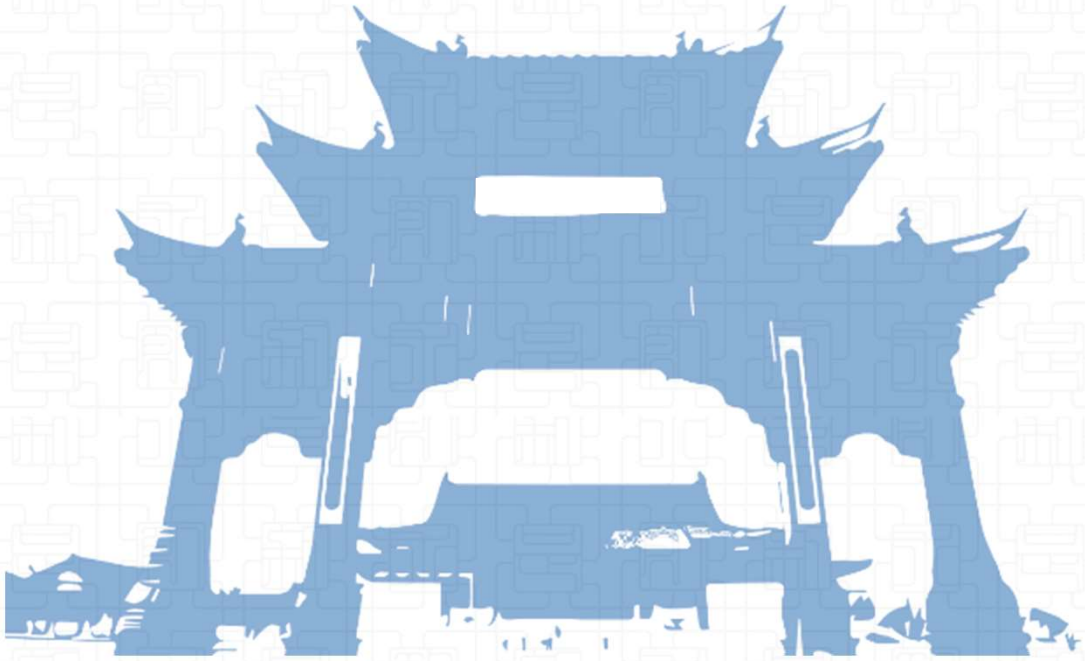
2.2 Method

哈爾濱工業大學 HARBIN INSTITUTE OF TECHNOLOGY

- For large demonstrations that exceed model input limits, we propose a divide-and-conquer (D\&C) aggregation method. This groups demonstrations into intermediate state vectors, which are then aggregated, allowing effective compression and representation of extensive ICL examples.



3 Experiments



3.1 Experimental Setting

哈爾濱工業大學 HARBIN INSTITUTE OF TECHNOLOGY

Optimization result

- Inner optimization method significantly improves the performance and robustness of the state vector compared to task and function vectors, reducing sensitivity to variations in demonstrations
- Momentum optimization further enhances the inner-optimized state vector, achieving top performance across settings and improving robustness, especially in zero-shot tasks.

Model		Method	Anym	Eng-Fr	Pers-Inst	Pers-Occ	Prod-Comp	Land-Cout	Average
Llama-2	Zero-shot	Regular	1.0± 0.2	0.1±0.1	0.0±0.0	0.0±0.0	0.4±0.2	0.0±0.0	0.3
		Function vector	45.1±2.0	21.6±2.0	11.3±10.7	0.1±0.1	25.6±4.3	32.9±21.6	22.8
		Task vector	56.2±2.8	63.2±3.6	61.8±8.4	27.9±15.2	55.5±20.1	57.8±26.3	53.7
		State vector (inn.)	61.0±1.0	66.5±2.2	67.4±2.6	42.7±4.2	64.5±10.6	81.0±1.7	63.9
		State vector (mom.)	60.4±0.7	67.5±1.8	68.7±1.6	45.6±5.9	71.3±3.6	77.7±1.8	65.2
	Few-shot	ICL baseline	64.8±4.8	74.3±0.8	71.7±3.7	56.1±2.7	80.8±0.8	87.0±0.3	72.5
		Function vector	54.5±0.9	65.2±1.4	60.8±5.6	54.2±2.2	76.0±1.3	84.2±2.9	65.8
		Task vector	65.7±1.8	73.8±0.9	66.6±5.2	56.4±2.3	81.9±1.8	86.7±0.9	71.8
		State vector (inn.)	**66.2**±1.6	**74.6**±0.9	70.1±4.3	57.0±2.2	**82.8**±1.6	87.5±0.9	73.0
		State vector (mom.)	65.8±3.7	74.3±1.1	**74.9**±2.9	**58.2**±0.4	82.0±1.0	**87.6**±0.3	**73.8**
GPT-J	Zero-shot	Regular	8.1±0.6	7.2±0.6	0.0±0.0	0.0±0.0	1.9±0.5	0.9±0.2	3.0
		Function vector	33.1±1.8	29.1±8.5	4.1±5.8	11.1±2.3	46.3±5.7	22.5±10.2	24.4
		Task vector	23.6±3.8	32.2±5.1	44.4±5.0	28.3±18.6	43.8±5.7	41.3±12.3	35.6
		State vector (inn.)	33.4±1.9	31.7±3.8	49.3±2.0	30.0±6.2	42.8±4.3	61.9±1.6	41.5
		State vector (mom.)	31.1±1.0	35.1±2.4	50.3±3.0	42.4±1.5	44.2±1.5	60.3±0.9	43.9
	Few-shot	ICL baseline	59.2±1.4	69.9±2.0	44.7±6.7	29.3±1.0	62.5±1.0	**69.3**±0.5	55.8
		Function vector	56.4±1.9	65.8±1.9	49.1±2.2	30.3±1.9	58.5±3.3	69.2±0.6	54.9
		Task vector	58.5±1.6	70.6±1.2	42.3±6.4	27.8±3.3	66.0±2.6	63.1±5.3	54.7
		State vector (inn.)	58.7±2.2	**70.9**±1.3	46.5±4.9	29.4±1.7	**66.3**±2.1	66.4±2.8	56.4
		State vector (mom.)	**59.6**±1.4	70.1±2.2	**51.9**±2.4	**30.4**±1.1	63.8±0.8	68.6±0.3	**57.4**

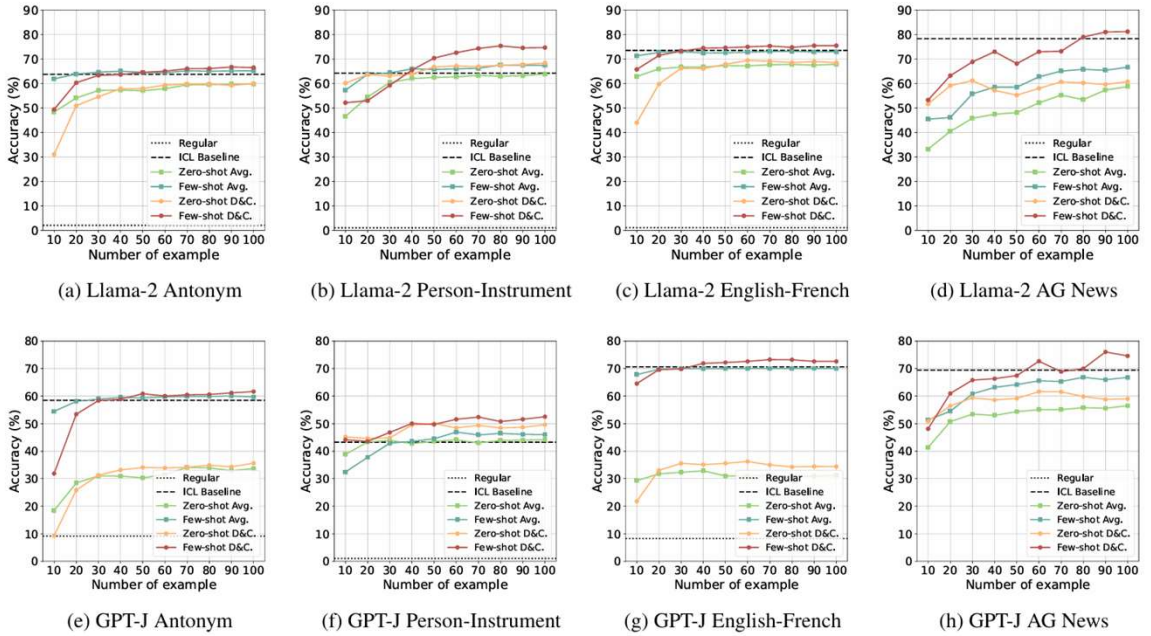
Table 1: Performance of state vector optimization. The best results in the zero shot setting are in underline and the best results in the few shot setting are in **bold**. The result of basic state vector is mathematically equivalent to task vector. Note that we only present the results across six tasks here and leave the rest in the Appendix. We also report standard deviation and the results are passed with significance test ($p < .05$).

3.1 Experimental Setting

哈爾濱工業大學 HARBIN INSTITUTE OF TECHNOLOGY

Aggregation result

- Our experiments demonstrate that both D\&C and average aggregation improve as the number of examples increases,
- D\&C aggregation surpasses average aggregation with multiple examples.



(a) Llama-2 Antonym (b) Llama-2 Person-Instrument (c) Llama-2 English-French (d) Llama-2 AG News

(e) GPT-J Antonym (f) GPT-J Person-Instrument (g) GPT-J English-French (h) GPT-J AG News

Figure 2: Performance of aggregation across number of examples. *Avg.* denotes the average aggregation baseline and *D&C.* denotes the divide-and-conquer aggregation. The **X** axis represents the number of examples, and the **Y** axis represents the accuracy.

3.1 Experimental Setting

哈爾濱工業大學 HARBIN INSTITUTE OF TECHNOLOGY

Ablation with Other Optimization Method

- Typical first-order gradient optimization are not as effective as momentum optimization

Method	Zero-shot	Few-shot
ICL baseline	0.2± 0.4	71.0± 10.8
Task vector	52.9± 9.4	68.5± 10.5
State vector (mom.)	65.2± 10.2	72.2± 10.6
State vector (adag.)	11.7± 12.0	16.1± 10.2
State vector (rms.)	0.8± 0.9	1.5± 1.0
State vector (adam.)	6.7± 6.1	10.6± 8.5

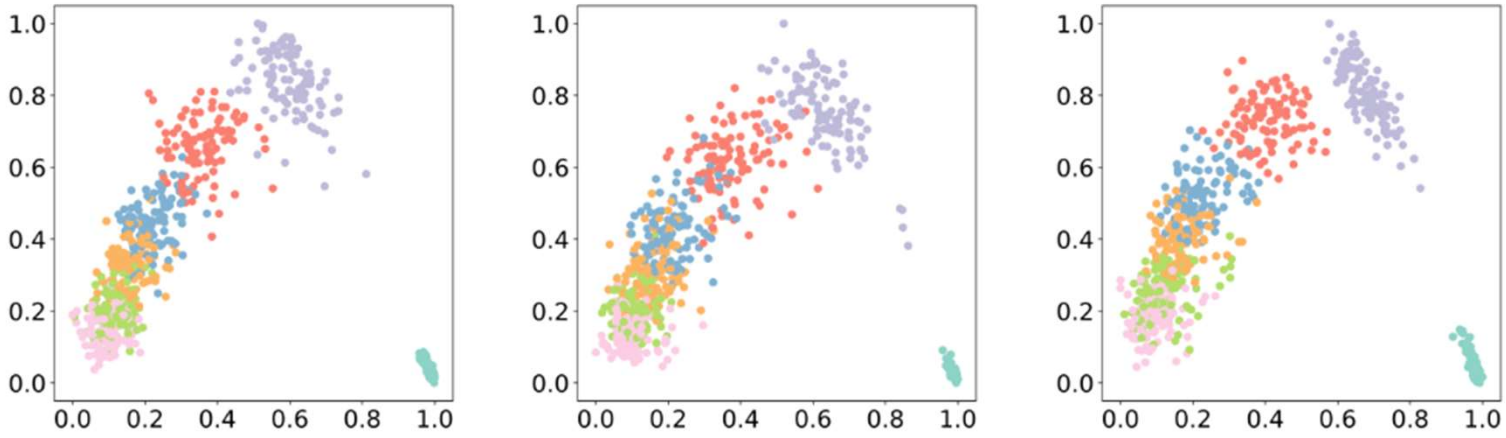
Table 2: Performance comparison of gradient optimization algorithms. The method means the optimization algorithm applied to the $\texttt{opt}(\cdot)$ in Eqn. 9.

3.1 Experimental Setting

哈爾濱工業大學 HARBIN INSTITUTE OF TECHNOLOGY

Qualitative Study

- State vectors corresponding to the examples occupying the same position tend to form distinct clusters.
- As the example position increases, clusters shift, suggesting the model gradually accumulates task-specific information.



(a) Antonym (b) English-French (c) Product-Company

4
Conclusion

4 Conclusion

哈爾濱工業大學 HARBIN INSTITUTE OF TECHNOLOGY

- In this paper, we reveal that ICL compressed vector can be viewed as parameters trained through gradient descent on the demonstrations. Then, we introduce the concept of state vector coupled with optimization and aggregation methods to enhance the capability of state vector and conduct comprehensive experiments across two popular LLMs and multiple tasks to support our claim. Our approach shows the ability to compress context while maintaining lower variance.

Thank You!

Email: liuzhenyuhit@gmail.com