HOW DOES REPRESENTATION IMPACT IN-CONTEXT LEARNING: A EXPLORATION ON A SYNTHETIC TASK

Jingwen Fu[1]*, **Tao Yang**[1]*, **Yuwang Wang**[2]†, **Yan Lu**[3] , **Nanning Zheng**[1]†
[1]Xi'an Jiaotong University, [2]Tsinghua University [3]Microsoft Reaserch Asia
jwfu99@gmail.com yt14212@stu.xjtu.edu.cn
wang-yuwang@mail.tsinghua.edu.cn yanlu@microsoft.com
nnzheng@mail.xjtu.edu.cn

ABSTRACT

In-context learning, i.e., learning from in-context samples, is an impressive ability of Transformer. However, the mechanism driving the in-context learning is not yet fully understood. In this study, we aim to investigate from an underexplored perspective of representation learning. The representation is more complex for in-context learning senario, where the representation can be impacted by both model weights and in-context samples. We refer the above two conceptually aspects of representation as in-weight component and in-context component, respectively. To study how the two components affect in-context learning capabilities, we construct a novel synthetic task, making it possible to device two probes, in-weights probe and in-context probe, to evaluate the two components, respectively. We demonstrate that the goodness of in-context component is highly related to the in-context learning performance, which indicates the entanglement between in-context learning and representation learning. Furthermore, we find that a good in-weights component can actually benefit the learning of the in-context component, indicating that in-weights learning should be the foundation of in-context learning. To further understand the the in-context learning mechanism and importance of the in-weights component, we proof by construction that a simple Transformer, which uses pattern matching and copy-past mechanism to perform in-context learning, can match the in-context learning performance with more complex, best tuned Transformer under the perfect in-weights component assumption. In short, those discoveries from representation learning perspective shed light on new approaches to improve the in-context capacity.

1 INTRODUCTION

Transformer-based models have demonstrated remarkable capacities for language processing (OpenAI, 2023; Devlin et al., 2018). Among the most astonishing aspects is their ability to rapidly learn from context examples (Brown et al., 2020), which is refered as in-context learning. With the advantage of no weight changeing, in-context learning has attracted a lot of research attentions and has been emploied to tackle real-world issues efficiently. This trend necessitates a deeper understanding of the underlying mechanisms behind in-context learning. Many efforts has been spent on this important topic. For example, several recent works (von Oswald et al., 2022; Dai et al., 2023; Akyürek et al., 2022) have described in-context learning as a form of gradient descent. Other works Li et al. (2023); Bai et al. (2023) further interpret in-context learning as algorithm implementation and selection.

In this paper, we head for exploring in-context learning from the aspective of representation learning, since it can often provide a deep understanding on how the internal representation related to the final output of the models. However, most previous works on investigating the representation within Transformer are outside of in-context learning senario (Schouten et al., 2022; Voita et al., 2019; Li et al., 2022) . The question that *how does the representation learning impact the in-context ability* is still underexplored.

*Equal contribution. Work done during internships at Microsoft Research Asia.
†Corresponding author

To answer this question, we first investigate the representation in the case of in-context learning. **The representation is actually impacted by not only the model weights but also in-context samples without weights update.** We thus conceptually decompose the representation into in-weight component and in-context component, related to in-weight learning (Chan et al., 2022b;a)) and in-context learing, respectively. We use *component* referring to PCA (Princeple Component Analysis). Here by conceptually decomposing, we do not mean explicitly decompose a single represention into two in-weight component and in-context component, phisically. Instead, we devise two probes, in-weight probe and in-context probe, to evaluate in-weight component and in-context component, respectively.

1.1 Main conclusions

With the above settings, we explore how the in-weight component and in-context component impact the in-context leaning capability. The experimental results reveal that: (i) Good in-weight component is a nessessary but not sufficent condition for strong in-context ability. (ii) The in-context component probing score is highly related to the in-context ability. However, it is hard to achieve strong in-context ability by only improving in-context component without considering in-weight conponent. (iii) By synergistically improving in-weight and in-context component, the in-context performance of a relative large model is lifted by a larger margin (in-context learning score rising from 0.168 to 0.885), and the in-context ability emergence is observed for a small model, as shown in Figure 1.

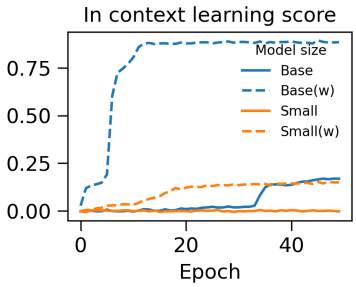

Figure 1: Test results for different training epoch and model size. Improving in-weights component(dashed line) can enable the emergence of in-context learning for small models. Detail in Section 3.2

The mathematical analysis is given to understand how Transformer perform in-context learning. We proof by construction that a simple Transformer can obtain a comparable performance with best trained more complex Transformer model(GPT2) in experimental part with only three layers under the assumption that the in-weights component is perfect learned in the input. This further verifies the important of in-weights components. Our construction reveals that pattern matching, as a supplement to copy-past mechansim(Edelman et al., 2022), is important for learning "sentence semantic" from in-context. The pattern matching refers to the mechanism that the Transformer will compare the patterns in different part of "image sentence" to obtain a global representation.

1.2 Contributions

Our main contributions can be summarized as following:

- From a representation learning perspective, we propose a new synthetic task to enable the study of the impact of *in-context component* and *in-weights component* on in-context ability seperately.
- The experimental results uncover the relationship between the quality of representations and in-context learning ability.
- Mathematical analysis is given to further understand the the in-context learning mechanism and importance of in-weights component.

2 Experimental Design.

In this section, we will talk about the experimental design, including dataset construction, model and training object, and the exploration framework.

Principles for Experimental Design 1) The prediction of the query example should adapt to the in-context example. 2) The evolution of in-weights and in-context components can be tracked. 3) The learning of in-weights and in-context components should be controllable.

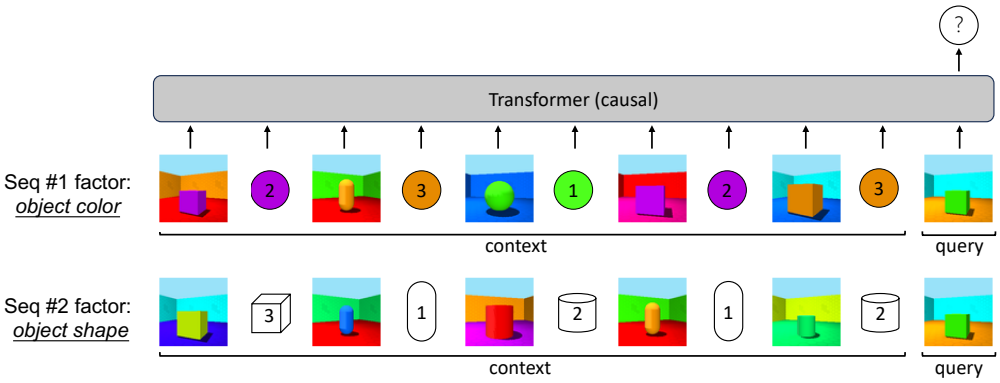

Figure 2: Experimental setup. We train Transformers by given sequence of image and lable pairs and training the model to predict labels of each image. Then, during inference, we evaluate the model's ability to predict accurately on new, unseen sequence. The images from 3D Shapes dataset are synthesized from six factors. The output factor is determined by the context. Here, we give the two sequences of factor "object color" and "object shape", respectively.

2.1 Dataset Construction

In general tasks, the impacts of model weights and context are entangled, and it is hard to design the above probes. Thus we propose a task on Shapes3D (Kim & Mnih, 2018) dataset to enable the study of in-weight an in-context components with two probes, respectively. The experimental setting is shown in Fig. 2. Specifically, given a sequence of image and label pairs as context, the task is to predict the label of the query image. For each image, there are six different factors: object color, object shape, object scale, background color, floor color and pose. We denote the factor as e and factor value of factor e as $V^{(e)}$ For each sequence, we randomly choose a factor to generate the labels of the images. We refer this factor as **hidden factor** for this sequence. For the two context seqences in Fig. 2, the factor of Seq #1 is object color, and the correct label for the qurey image is 1 (object color is green). In Seq #2, for the same query image, the correct label is 3 (object shape is cube).

To make the correct prediction, the network needs not only to recognize the values of six factors from the query image (related to in-weight component), but also to identify the correct factor to output from the contexts (related to in-context component). Therefore, the in-weight probe is to predict the values of six factors, only given the query image. The in-context probe is to identify the factor of the context, given the full sequence.

To study the relationship between how well the in-weight/context components is learnt and how strong the in-context learning ability is, we **control the difficulty in learning the representation components by designing two label assignment settings** during training phase:

- D_{fix}: the mapping between labels and factor values is fixed across all the sequence.
- D_{rnd}: the mapping is fixed only inside one single sequence, and randomly shuffled for different sequences.

The model is expected to better learning in-weights component in D_{fix} setting and is expected to learn better in-context component in D_{rnd}. To further analysis the interplay between these components, we further consider the two composite setting:

- $D_{\text{fix}\to\text{rnd}}$: In this setting, we first train the model on D_{fix} for a specific epoch, and then, we train the model on D_{rnd}.
- $D_{\text{fix}\wedge\text{rnd}}$: The 50% data in the training set uses D_{fix} and the rest uses D_{rnd}.

Given two data settings D_1 and D_2, we use $D_1 \Rightarrow D_2$ to denote the evaluation result of model on data setting D_2 after the model is trained on data setting D_1. Fig. 3 gives the correspondings between the relations we want to explore within in-weights components, in-context components and in-context learning. When tesing the in-context ability during inference, we prefer the D_{rnd} setting,

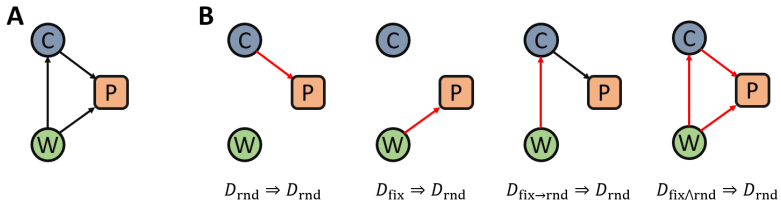

Figure 3: Illustration of task design. C, W and P are in-context component, in-weights component and in-context performance of Transformer respectively. A: All possible relations. B: The relations that we want to explore in each experiments setting. The red line denotes that the explored relation and the black line denoted that the non-removed relations.

referring to Wei et al. (2023); Min et al. (2022), which point out that the label shufffuled case can better distinguish in-context ability.

2.2 Model and Training

Based on the proposed task, we want to explore the properties of Transformer. We leverage the causal Transformer, where every token can only attend to the other tokens previous to it. Without specific, we implement the Transformer f as GPT2 model with 12 layers, 4 heads and embedding size 128. To simulate the auto-regression training framework, we calculate the loss for the sequence $s = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_L, y_L)\}$ as:

$$\mathcal{L}(\theta, s) = \frac{1}{L} \sum_{i=1}^{L} l(f(\{\mathbf{x}_1, y_1, \cdots, \mathbf{x}_i\}), y_i), \tag{1}$$

where l denotes the loss function. x will be tokenized by VAE before being passed to Transformer. The training loss in the dataset S is calculated as the average of loss over all training sequence, i.e.,

$$\mathcal{L}(\theta, S) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(\theta, s_i). \tag{2}$$

Here, we leverage the Adam(Kingma & Ba, 2014) and mini-batch training to optimize Loss(θ, S). The batch size used here is 128 and the learning rate is set to 0.0001.

2.3 Exploration Framework

Here we give the metrics to give numeral evaluation of the components, and in-context learning performance. Since the components are hidded in the representation, we rely on the probe method (Alain & Bengio, 2016). The probe classifier consists of a single linear layer, and the loss is calculated using softmax and cross-entropy. The probe classifer is trained for one epoch.

In-context comp. score (Fig. 4A) The in-context comp. score is calculated as $\frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{\hat{e}_i = e_i}$, where $\mathbf{1}_{\text{expr}}$ is 1 when the expr is true else it is 0. e_i is the hidden factor of the i-th "image sentence" and $\hat{e}_i$ is the prediction of probe classifier. We use the e to evaluate, because e can only be learned through in-context learning.

in-weights comp. score (Fig. 4B) To remove the influence in-context components, we disable the attention layer in the Transformer. Then the in-weights comp score is calculated as $\sum_{i=1}^{n}\sum_{j=1}^{L}\sum_{k=1}^{|E|} \mathbf{1}_{\hat{v}_k^{(E_k)} = v_k^{(E_k)}}$, where $\hat{v}_k^{(E_k)}$ is the prediction of probe classifier.

In-context learning score (Fig. 4C) Let a_i and a_j be the accuracies of the Transformer on the i-th and j-th samples, respectively. Following Olsson et al. (2022), the in-context learning score is calculated as $a_i - a_j$. Here we choose $i = 40, j = 0$ by defailt.

Rationale to leverage hidden factor prediction as probe task. To solve the task on D_{rnd}, we need to obtain two informations from in-context, that is the hidden factors and the mapping between

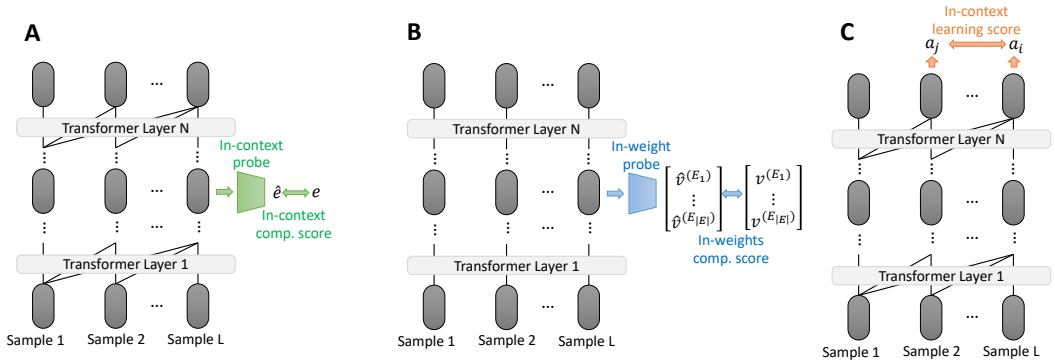

Figure 4: Illustration of calculation of in-context comp. score(A), in-weights comp score(B) and in-context learning score(C).

factor values and labels. Therefore, we can only choose the probe task for in-context component from these two candidates. **The mapping of the factor values and labels can not be used as probe task** because 1) the number all possible mapping is much larger than the size of dataset, which means we cannot learn the probe classifier. 2) The information of mapping is not neccessary stored in representation. Section 3.3 gives a solution of the model that no information of mapping stored in representation. **Hidden factors prediction is suitable for probe task** because 1) Hidden factor is the sequence level information that can only be learned from in-context example. 2) It is neccesary for solving the tasks and its information is stored in representation (Section 3.1).

2.4 COMPARISON OTHER EXPLORATION WITH SYNTHETIC TASKS

Table 1: Comparison with other papers that explore in-context learning using synthetic dataset.

	Garg et al. (2022)	Chan et al. (2022a)	Ours
Synthetic task	Simple functions	Image data	Image data
Sentence Semantic	No	No	Yes
Perspective	Algorithm implementing	Data properties	Representation
In-weights learning	No	Trade off with in-context learning	Complex relations with in-context learning

There are two kinds of synthetic tasks are common used in the exploration of in-context learning:

- (ST1, simple functions) In this task, a simple function is sampled for each sentence (an input sequence for the Transformer). Then, x_i is generated by sampling from a specific distribution, and y_i is produced using the sampled simple function with x_i as input.
- (ST2, image sequence) In this task, x_i is a randomly sampled image from the image dataset, and y_i is generated using the original label values.

(ST1) is investigated in the works by Garg et al. (2022); von Oswald et al. (2022); Akyürek et al. (2022). (ST2) is examined in the studies by Chan et al. (2022a); Kirsch et al. (2022); Chan et al. (2022b). (ST1) researches in-context learning at a more abstract level, leading to the conclusion that in-context learning implements algorithms, such as gradient descent(von Oswald et al., 2022). However, their tasks are significantly distant from real applications because 1) The input token x consists of numbers without any evident pattern or semantics, while most tokens in NLP tasks are words with clear meanings. 2) The tangible forms of their results is hard to be found in practice. The resolution of real NLP tasks is difficult to be expressed as straightforward, comprehensible algorithms, such as gradient descent or ridge regression. In contrast, (ST2) is closer to real applications since the image data used has semantic meaning. Thus, it is feasible to investigate how data properties influence in-context learning (Chan et al., 2022a). Our synthetic task belongs to the (ST2) category. The primary distinction between our synthetic task and the previous tasks in (ST2) is that "sentence semantics" are considered in our approach. In the following, we provide a detailed comparison between our work and that of Chan et al. (2022a).

Comparison with Chan et al. (2022a) **Regarding dataset setting:** The primary difference in our task setting compared to that of Chan et al. (2022a) lies in the consideration of "sentence semantics." Specifically, if we remove certain factors or only consider one factor, our constructed synthetic data would degenerate to that of Chan et al. (2022a). The **importance** of considering "Sentence Semantic" lies in the following: 1) Understanding sentence semantics plays a crucial role in practical applications, as evidenced by various studies (Zheng et al., 2021; Bowerman, 1976; Reimers & Gurevych, 2019). 2) Without sentence semantics, the function of in-context learning would degenerate into a simple copy-paste mechanism, wherein the Transformer can predict the label of a query image by searching for context images with the same label and then copying the label of the in-context image to the prediction of the query image. **Regarding the division of in-weights/in-context:** At a high level, the meanings of "in-context" and "in-weights" are consistent across our work and that of Chan et al. (2022a). However, our paper advances further by: 1) Analyzing the complex relationship between in-weights and in-context components from a representation perspective, leading to a more realistic conclusion. Chan et al. (2022a) concludes that in-context learning and in-weights learning are in a tradeoff relationship in their exploration, but large language models can exhibit both capacities Brown et al. (2020), which is acknowledged by Chan et al. (2022a) in the discussion at the end of their paper. 2) While Chan et al. (2022a) devises two tasks to evaluate in-context and in-weights learning, our work leverages a task that requires both in-context and in-weights information to solve. This setting is more closely aligned with practical applications, as real tasks often require both types of information (Brown et al., 2020; Alayrac et al., 2022).

3 RESULTS

In this part, we give our experimental findings. In section 3.1, we will explore how in-weights component and sequence representation contribute to the in-context learning ability individually. In section 3.2, we will explore the joint effect of in-weights component and in-context component. Finally, we want to explore how Transformers solve this problem in section 3.3

3.1 INDIVIDUAL IMPACT OF IN-CONTEXT COMPONENT AND IN-WEIGHTS COMPONENT

Key points We design experiments to **solely** improve in-context component or in-weight component, we find that **1) the in-context component is highly related to in-context learning and 2) solely improve in-weights component cannot affect in-context learning perfomance.**

To verify the effective of task deign, we first conduct experiments to explore the evolving of representation learning when applied data setting D_{fix} and D_{rnd}. We find that

- **D_{rnd} struggles to learn effective in-weights component, while D_{fix} excels in this task.** We evaluate the in-weights component of D_{fix} and D_{rnd} during the training process in Fig. 5A. The in-weights component quality of D_{fix} significantly surpasses that of D_{rnd}. And during training, the in-weights component of D_{fix} improves, while the in-weights component of D_{rnd} fluctuates below the initial level.
- **The in-context learning components learned from D_{rnd} can be applied to D_{fix}, but the reverse does not hold true.** When trained on D_{rnd},$D_{\text{rnd}} \Rightarrow D_{\text{rnd}}$ and $D_{\text{rnd}} \Rightarrow D_{\text{fix}}$ have a similar in-context comp score in Fig. 5A. On the other hand, model trained on D_{fix} shows exciting in-context comp score when evaluate on D_{fix}, but on D_{rnd}, we cannot observe a obvious increase in in-context comp score.

Based on the information of transfer ability of different components between D_{rnd} and D_{fix}, we further explore the problem of concern that how the representation impacts in-context learning. The following observations are obtained,

- **In-context component is high correlated with the in-context learning ability.** In Fig. 5B, we evaluate the in-context components and the in-context learning performance during the training process with different task settings. In all these setting, the in-context comp score has a similar trend with the in-context learning performance. The results indicate that in-context learning of Transformer relies on its inner representation and the increase of the in-context learning performance may come from the representation learning.

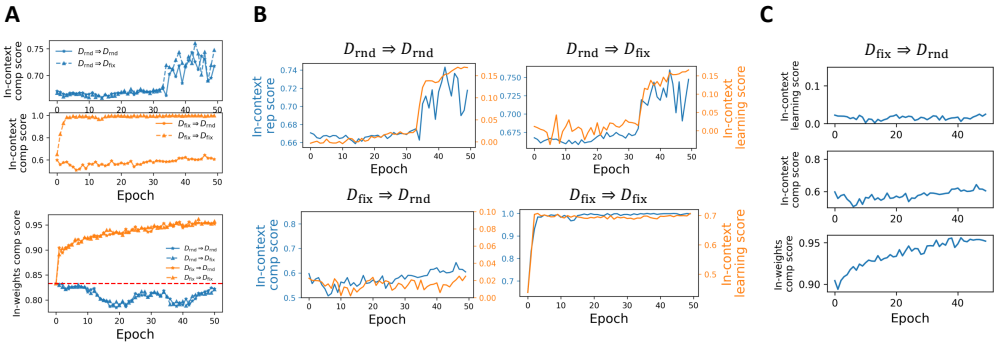

Figure 5: Investivation of in-context representation, in-weights component and in-context learning performance in different data settings. A: The in-context learning ability learned from D_{rnd} can be applied to D_{fix}, but the reverse does not hold true. And the in-context learning representation is highly related to the in-context learning performance. B: The in-weights representation will diminish when trained on D_{rnd}, while it will enhance when trained on D_{fix}.

- **In-weights component cannot direct impact the in-context learning performance.** We take a closer look at $D_{\text{fix}} \Rightarrow D_{\text{rnd}}$, and the results are given in Fig. 5C. Because the in-context component learned from D_{fix} cannot transfer to D_{rnd}, only the in-weights component is improved. No obvious improvement is observed and this means that in-weights component doesn't dirrectly affect the in-context learning performance.

3.2 Joint impact of in-context component and in-weights component

Key points We explore the **cooperation** between in-context component and in-weights component, and we find that **1) Good in-weights component can help the learning of in-context component 2) Simultaneously improve in-weights component and in-context component is more effective.**

We consider the setting $D_{\text{fix}\to\text{rnd}} \Rightarrow D_{\text{rnd}}$. The model is first trained on the D_{fix} to improve the in-weights component and then we transfer to D_{rnd} to improve its in-context component. In this way, we can explore how the in-weights component affects the in-context components. We observe that:

- **Better in-weights component can accelerate the learning of in-context component** In Fig. 6A ($D_{\text{fix}\to\text{rnd}} \Rightarrow D_{\text{rnd}}$), we observe a sudden increase of in-context comp score at the point when we switch from D_{fix} to D_{rnd}. This results indicate that if we can quicker learn the in-context component based on a representation with better in-weights component.

Then, we further explore the collective effect of in-weights and in-context components on in-context learning by simutainously improve the in-weights and in-context components using the task setting $D_{\text{fix}\wedge\text{rnd}} \Rightarrow D_{\text{fix}}$. We find that

- **Learning in-weights component and in-context component simultaneously is more effective than learning them separately.** Compared with training on $D_{\text{fix}\Rightarrow\text{rnd}}$, model trained on $D_{\text{fix}\wedge\text{rnd}}$ learns much faster. What's more, $D_{\text{fix}\wedge\text{rnd}}$ can achieve better in-context performance with similar in-context comp score, which indicates that learning in-weights components and sequence representations simultaneously is more powerful.

3.3 How Transformer solve this problem?

Key points We explore the **mechanism** of in-context learning by construction. We find 1) **The importance of in-weights component conclude in the experiments is further verified** by the fact that a simple Transformer can learn powerful in-context learning results based on perfect in-weights component assumption. 2) **The Transformer may rely on pattern matching and copy-past mechanism** for in-context learning.

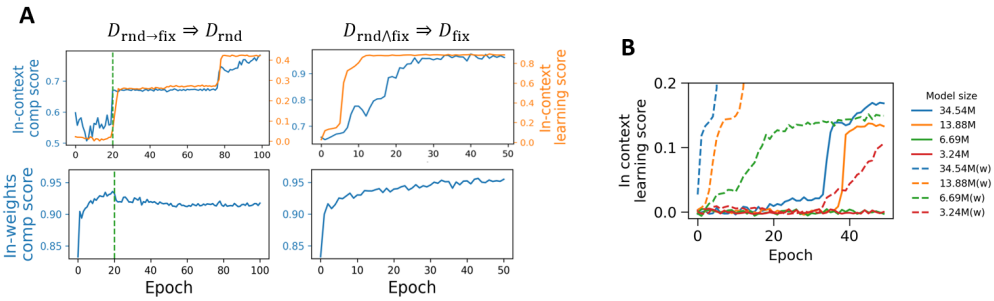

Figure 6: Improve in-weights component and in-context component by combining D_{fix} and D_{rnd}. A: Better in-weights component can accelerate the learning of in-context component. The green dash line denotes point when switch from D_{fix} to D_{rnd}. B: Improving in-weights component can enable the emergency of in-context learning ability in small model. The dashed line denotes the task setting $D_{\text{fix}\wedge\text{rnd}} \Rightarrow D_{\text{rnd}}$ while the solid line denotes the setting $D_{\text{rnd}} \Rightarrow D_{\text{rnd}}$.

We consider the naive Transformer(Vaswani et al., 2017). The hidden representation of token i in Transformer is denoted as $\mathbf{h}_i \in \mathbb{R}^d$. The whole hidden state of the sequence is denoted as $\mathbf{H} = [\mathbf{h}_1, \cdots, \mathbf{h}_{2L}]^{\mathrm{T}} \in \mathbb{R}^{2L\times d}$. The hidden representation of l-th layer is denoted as $\mathbf{H}^{(l)}$.

Definition 3.1. *(Transformer) One layer of Transformer contains one attention layer and one MLP layer. The calculation of Attention Layer is*

$$\mathrm{Attn}^{(l)}(\mathbf{H}^{(l)}) = \mathbf{H}^{(l)} + \sum_{c=1}^{C} \sigma\left(\mathbf{H}^{(l)}\mathbf{W}_Q^{(l,c)}(\mathbf{H}^{(l)}\mathbf{W}_K^{(l,c)})^{\mathrm{T}}\right)\mathbf{H}^{(l)}\mathbf{W}_V^{(l,c)}\mathbf{W}_O^{(l,c)}. \tag{3}$$

And the calculation of MLP layer is

$$\mathbf{H}^{(l+1)} = \mathrm{Attn}^{(l)}(\mathbf{H}^{(l)}) + \mathbf{W}_2^{(l)}\,\mathrm{Relu}(\mathbf{W}_1^{(l)}\,\mathrm{Attn}^{(l)}(\mathbf{H}^{(l)})). \tag{4}$$

Here we consider relaxed case where $\sigma = \mathrm{Id}$.

The relaxion of Transformer is discussed by many previous works. Press et al. (2019) discover using the Relu in fead forward layer can acheiver comparable results in original one. Wiegreffe & Pinter (2019); Brunner et al. (2019); Richter & Wattenhofer (2020) points that softmax operation may not actually needed for Transformer.

Definition 3.2. *(Perfect in-weights component) If feature* h *has a perfect in-weights component, then for all factor* e*, exists* $\mathbf{W}_e \in \mathbb{R}^{d\times|V_e|}$ *such that* $\mathbf{f}^{(e)}_{\mathbf{x}_1} \cdot \mathbf{f}^{(e)}_{\mathbf{x}_2} = 1$ *only when* $v^{(e)}_{x_1} = v^{(e)}_{x_2}$*, else we have* $\mathbf{f}^{(e)}_{\mathbf{x}_1} \cdot \mathbf{f}^{(e)}_{\mathbf{x}_2} = 0$.

Based on the perfect in-weights component assumption, we can construct a Transformer with additional three layers to learn the in-context component and achieve comparable performance compared with the best trained GPT2 model in previous experiments.

Proposition 3.3. *We consider the data with* n_e *factors and each factor has* n_v *values in* D_{rnd} *setting. For causal Transformer with the number of heads larger or equal the number of factors with the hidden size* $\mathcal{O}(n_e n_v + L)$*, if the Transformer can learn a perfect in-weights component in layer* k*, then it can learn a feature given* i *in-context samples with sequence representation score* $\mathrm{srs}_i = (1 - \mathrm{srs}_{i-1})s_i + \mathrm{srs}_{i-1}$ *and* $\mathrm{srs}_0 = s_0$ *at layer* $k+2$*, where* $s_i = 1 - \sum_{j=0}^{i}\binom{i}{j}\sum_{k=2}^{|E|}\binom{|E|}{k}\frac{k-1}{k}\left(\frac{(n_v-1)^{i-j}}{n_v^i}\right)^k\left(1 - \frac{(n_v-1)^{i-j}}{n_v^i}\right)^{|E|-k}$*, and we can obtain in-context learning score as* $\mathrm{cls}_i = \frac{(n_v-1)(n_v^{i-1}-(n_v-1)^{i-1})}{n_v^i}\,\mathrm{srs}_i$ *at* $k+3$ *layers.*

Proof Sketch The constuction of weights can be divided into two steps: **1. Estimate the factor in this sequence**.According to the perfect in-weights component assumption, we can project the token feature into the space $\mathbf{f}_e$. The factor is chosed by find e such that $(\mathbf{f}_e^i)^{\mathrm{T}}\mathbf{f}_e^j$ can match $y_i^{\mathrm{T}}y_j$ **2. Estimate** y**.** Based on the discovered factor in previous step, we choose the corresponding $\mathbf{W}_e$ to map the distribution in its corresponding space and assign the label of new sample to the label of its nearest sample.

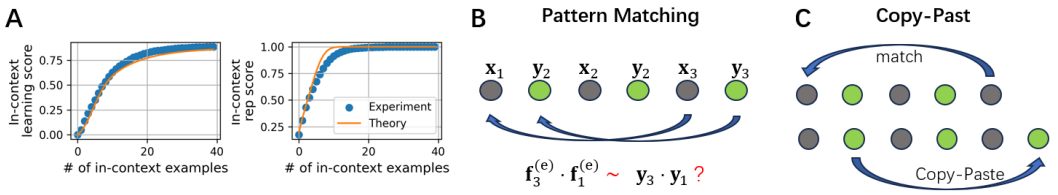

Figure 7: A: The constructed Transformer can match the performancce of best trained GPT2. B: Two mechanisms for the constructed Transformer to learn in-context.

The in-context learning can be easily obtained based on a good in-weights component Fig. 7 shows that the performance of constructed simple Transformer can match the best tuned GPT2 model under the perfect in-weights component assumption. Due to the limited capacity of three simple Transformer layers, we can infer that the in-context learning can be easily obtained based on a good in-weights component.

Pattern matching and copy-past may be a possible mechansim for in-context learning. The constructed Transformer relies on two mechanism (shown in Fig. 7B) to obtain in-context information. The first is pattern matching to learn the hidden factor of each image sentence. The pattern matching compare the product value between the factor feature $\mathbf{f}^e$ and labels to determine which factor generates the labels in this sequence. Another mechanism is copy and past. Instead to learn the mapping between factor value and labels, the Transformer simply copy the label of in-context sample with a same value of hidden factor. The similar performance between the constructed Transformer and the best trained GPT2 model implies that the two mechanism may applied in trained GPT2. Olsson et al. (2022) also finds that copy-past mechanism plays an important role for in-context learning.

Comparison with previous work in analyzing the in-context learning mechanism The **key contribution** of our analysis is to analysis how the in-context learning learn the "sentence semantic",i.e. hidden factor, from in-context samples. Previous works investigate the mechanism of in-context learning mainly focus on the pair-wise relation between the query tokens and in-context tokens, and they reveals that copy-past is a important mechanism for in-context learning. In this paper, we want to analyze how Transformer learn the "Sentence Semantic". In our construction, we find that model may rely on the comparison of information within "Sentence" (i.e., pattern matching).

4 Limitation

This paper investigates the relationship between representation and in-context learning using a synthetic dataset. Although some discrepancies exist between our synthetic task and real-world applications, directly exploring practical tasks is challenging due to their complexity and uncontrollability. Nonetheless, previous studies on synthetic tasks have provided valuable insights into in-context learnings(Garg et al., 2022; Chan et al., 2022a;b; von Oswald et al., 2022). Compared to previous synthetic tasks, our work is more closely aligned with practical applications, as discussed in Section 2.4.

5 Conclusion

This paper analyze the relation between representation and in-context learning by decomposite the representation into in-weights and in-context components. Our experiments reveal that the in-context component is directly related to the in-context learning ability. Further exploration reveal that in-weights component is essential for the learning of in-context component. The discoveries are further explored by constructing a simple Transformer to match the performance of best trained GPT2 model. In short, this paper reveals how representation impacts in-context learning under a synthetic dataset.

REFERENCES

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. *arXiv preprint arXiv:2306.04637*, 2023.

Alberto Bietti, Vivien Cabannes, Diane Bouchacourt, Herve Jegou, and Leon Bottou. Birth of a transformer: A memory viewpoint. *arXiv preprint arXiv:2306.00802*, 2023.

Melissa Bowerman. Semantic factors in the acquisition of rules for word use and sentence construction. In *Directions in normal and deficient language development*, pp. 99–179. University Park Press, 1976.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. On identifiability in transformers. *arXiv preprint arXiv:1908.04211*, 2019.

Stephanie Chan, Adam Santoro, Andrew Lampinen, Jane Wang, Aaditya Singh, Pierre Richemond, James McClelland, and Felix Hill. Data distributional properties drive emergent in-context learning in transformers. *Advances in Neural Information Processing Systems*, 35:18878–18891, 2022a.

Stephanie CY Chan, Ishita Dasgupta, Junkyung Kim, Dharshan Kumaran, Andrew K Lampinen, and Felix Hill. Transformers generalize differently from information stored in context vs in weights. *arXiv preprint arXiv:2210.05675*, 2022b.

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. Why can gpt learn in-context? language models implicitly perform gradient descent as meta-optimizers. In *ICLR 2023 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2023.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Benjamin L Edelman, Surbhi Goel, Sham Kakade, and Cyril Zhang. Inductive biases and variable creation in self-attention mechanisms. In *International Conference on Machine Learning*, pp. 5793–5831. PMLR, 2022.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems*, 35:30583–30598, 2022.

Angeliki Giannou, Shashank Rajput, Jy-yong Sohn, Kangwook Lee, Jason D Lee, and Dimitris Papailiopoulos. Looped transformers as programmable computers. *arXiv preprint arXiv:2301.13196*, 2023.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *International Conference on Machine Learning*, pp. 2649–2658. PMLR, 2018.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Louis Kirsch, James Harrison, Jascha Sohl-Dickstein, and Luke Metz. General-purpose in-context learning by meta-learning transformers. *arXiv preprint arXiv:2212.04458*, 2022.

Kenneth Li, Aspen K Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent world representations: Exploring a sequence model trained on a synthetic task. *arXiv preprint arXiv:2210.13382*, 2022.

Yingcong Li, Muhammed Emrullah Ildiz, Dimitris Papailiopoulos, and Samet Oymak. Transformers as algorithms: Generalization and stability in in-context learning. 2023.

Bingbin Liu, Jordan T Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Transformers learn shortcuts to automata. *arXiv preprint arXiv:2210.10749*, 2022.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*, 2022.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

OpenAI. Gpt-4 technical report. *arXiv*, 2023.

Ofir Press, Noah A Smith, and Omer Levy. Improving transformer models by reordering their sublayers. *arXiv preprint arXiv:1911.03864*, 2019.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

Oliver Richter and Roger Wattenhofer. Normalized attention without probability cage. *arXiv preprint arXiv:2005.09561*, 2020.

Stefan Schouten, Peter Bloem, and Piek Vossen. Probing the representations of named entities in transformer-based language models. In *Proceedings of the Fifth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pp. 384–393, 2022.

Yuandong Tian, Yiping Wang, Beidi Chen, and Simon Du. Scan and snap: Understanding training dynamics and token composition in 1-layer transformer. *arXiv preprint arXiv:2305.16380*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Elena Voita and Ivan Titov. Information-theoretic probing with minimum description length. *arXiv preprint arXiv:2003.12298*, 2020.

Elena Voita, Rico Sennrich, and Ivan Titov. The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives. *arXiv preprint arXiv:1909.01380*, 2019.

Johannes von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. *arXiv preprint arXiv:2212.07677*, 2022.

Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, et al. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*, 2023.

Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. *arXiv preprint arXiv:1908.04626*, 2019.

Wenfeng Zheng, Xiangjun Liu, and Lirong Yin. Sentence representation method based on multi-layer semantic network. *Applied sciences*, 11(3):1316, 2021.

A Dataset detail

3dshapes[1] is a dataset of 3D shapes procedurally generated from 6 ground truth independent latent factors. These factors are floor colour, wall colour, object colour, scale, shape and orientation.

All possible combinations of these latents are present exactly once, generating N = 480000 total images.

Latent factor values floor hue: 10 values linearly spaced in [0, 1] wall hue: 10 values linearly spaced in [0, 1] object hue: 10 values linearly spaced in [0, 1] scale: 8 values linearly spaced in [0, 1] shape: 4 values in [0, 1, 2, 3] orientation: 15 values linearly spaced in [-30, 30] We varied one latent at a time (starting from orientation, then shape, etc), and sequentially stored the images in fixed order in the images array. The corresponding values of the factors are stored in the same order in the labels array.

B Other related work

We discuss the most related work in the main part of paper. Here, we list other works that are weaker related to us.

Analysis of Transformer The analysis of Transformers can be broken down into two main components: examining the expressibility of Transformers and comprehending the mechanisms of learned Transformers. To analyze the expressibility of Transformers, a common approach is to determine if they can solve specific problems by constructing appropriate weights. Giannou et al. (2023) demonstrates that Transformers can function as Turing machines, while Liu et al. (2022) shows that they can learn shortcuts to solve automata problems. In addition to expressibility, researchers have also investigated the mechanisms behind learned Transformers. Bietti et al. (2023) examines Transformers from a memory standpoint, and Tian et al. (2023) focuses on single-layer Transformers. While the analysis of Transformers is crucial to our work, our ultimate goal differs; we aim to bridge the gap between representation learning and in-context learning.

Exploration of representation within Transformer. Owing to the widespread use of Transformers, numerous studies (Li et al., 2022; Voita & Titov, 2020) seek to investigate their internal representations as a means of comprehending their functionality. The most prevalent approach involves utilizing probe models and tasks to discern the information stored within these representations (Voita & Titov, 2020; Schouten et al., 2022). Taking a different perspective, Voita et al. (2019) explores the flow of information across Transformer layers and how this process is influenced by the selection of learning objectives. Our work shares similarities with these studies in that we employ the probe method to examine representations. However, our focus differs in that we do not concentrate on the semantic meaning within the representation. Instead, we investigate how the in-weights and in-context information impact representation.

C Proof of Proposition C.3

We consider the position embedding as $\mathbf{p}_i = (0, \cdots 0, 1, 0, \cdots)^{\mathrm{T}}$, where we only have value 1 at i-th position and 0 others.

Lemma C.1. *The attention module can implement copy and past behavior.*

Proof. According to the definition of $\mathbf{p}_i$, we have $\mathbf{p}_i \cdot \mathbf{p}_j = 0$ if $i \neq j$, otherwise, we have $\mathbf{p}_i \cdot \mathbf{p}_j = 1$. We denote

$$\mathbf{M} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}.$$

[1] https://github.com/deepmind/3d-shapes

Then we have $\mathbf{p}_i\mathbf{M} = \mathbf{p}_{i-1}$. For $j > i$, we denote the value of j-th token as $\mathbf{h}_j = (\mathbf{0}, \mathbf{p}_j)$ and i-th token as $\mathbf{h}_i = (\mathbf{h}'_i, \mathbf{p}_i)$. If we want to copy the value of i-th token to the value of j-th token, we can set the query matrix as $\mathbf{W}_Q = (\mathbf{0}, \mathbf{M}^{j-i})$, the key matrix as $\mathbf{W}_K = (\mathbf{0}, \mathbf{I})$ and value matrix as $\mathbf{W}_V = (\mathbf{W}'_V, \mathbf{0})$. Then we have

$$\mathbf{h}_j^{\mathrm{T}}\mathbf{W}_Q \cdot \mathbf{h}_a^{\mathrm{T}}\mathbf{W}_K = \mathbf{p}_i \cdot \mathbf{p}_a = \begin{cases} 1 & a \neq j \\ 0 & a = j \end{cases} \tag{5}$$

Therefore, the j-th token can only attend to i-th token. Then we have the value of $\mathbf{h}_j$ after attention as $\mathbf{h}_j^{attn} = ((\mathbf{h}'_i)^{\mathrm{T}}\mathbf{W}_V, \mathbf{p}_j)$. By setting $\mathbf{W}_V$ as different value, we can copy different part information of i-th to j-th token. Then the lemma is held. □

Lemma C.2. *For the input $h = [h_1, h_2, h_3]$, where $h_i \in \mathbb{R}^{d_i}$ and $d_1 + d_2 + d_3 = d$, for all $MLP_s(h) = W'_2 Relu(W'_1 h_2) : \mathbb{R}^{d_2} \to \mathbb{R}^{d_2}$, there exists $MLP_s(h) = W_2 Relu(W_1 h) : \mathbb{R}^d \to \mathbb{R}^d$, such that $MLP(h) = [h_1, MLP_s(h_2), h_3]$.*

Proof. Obviously, for any W'_1, there exists W_1, such that $W_1\ h^{(1)} = W_1 h = [h_1, -h_1, W'_1 h_2, h_3, -h_3]$.

Obviously, for any W'_2, There exists W_2, such that $h^{(2)} = W_2 Relu(h^{(1)}) = [Relu(h_1) + Relu(-h_1), W'_2 Relu(W'_1), Relu(h3) + Relu(-h3)] = [h_1, MLP_s(h_2), h_3]$ □

Proposition C.3. *We consider the data with n_e factors and each factor has n_v values in D_{rnd} setting. For causal Transformer with the number of heads larger or equal the number of factors with the hidden size $\mathcal{O}(n_e n_v + L)$, if the Transformer can learn a perfect in-weights component in layer k, then it can learn a feature given i in-context samples with sequence representation score $\mathrm{srs}_i = (1 - \mathrm{srs}_{i-1})s_i + \mathrm{srs}_{i-1}$ and $\mathrm{srs}_0 = s_0$ at layer $k + 2$, where $s_i = 1 - \sum_{j=0}^{i} \binom{i}{j} \sum_{k=2}^{|E|} \binom{|E|}{k} \frac{k-1}{k} \left(\frac{(v-1)^{i-j}}{v^i}\right)^k \left(1 - \frac{(v-1)^{i-j}}{v^i}\right)^{|E|-k}$, and we can obtain in-context learning score as $\mathrm{cls}_i = \frac{(v-1)(v^{i-1} - (v-1)^{i-1})}{v^i} \mathrm{srs}_i$ at $k + 3$ layers.*

Proof. Without loss of generality, we assume the representation of Transformer in layer k is in a form that $\mathbf{h}_{2i-1}^{(k)} = (\mathbf{f}_i, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{p}_i^{\mathrm{T}})^{\mathrm{T}}$ and $\mathbf{h}_{2i}^{(k)} = (\mathbf{0}, \mathbf{l}_i, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{p}_i^{\mathrm{T}})^{\mathrm{T}}$. Because the representation usually lays in low dimension space, a simple linear layer can transfer the representation in our defined sparse form. What's more, it is nature to assume that the position information is stored in the representation, since it is given in the input and it is essential for attention.

The consider the operations of Transformer in different layers.

Layer 1

Because we assume that $\mathbf{h}_{2i-1}^{(k)}$ is a perfect token representation, then there exists $\mathbf{W}_e$, such that $\mathbf{h}_{2i-1}^{(k)}\mathbf{W}_e = \mathbf{f}_i^e$, where $\mathbf{f}_i^{(e)}$ satisfies that $\forall e, i$, we have $\mathbf{f}_j^{(e)} \cdot \mathbf{f}_i^{(e)} = 1$ only when $v_i^{(e)} = v_j^{(e)}$ else $\mathbf{f}_j^{(e)} \cdot \mathbf{f}_i^{(e)} = 0$.

Then, assign $\mathbf{W}_Q^{(l,i)} = \mathbf{W}_K^{(l,i)} = \mathbf{W}_{e_i}$ and $\mathbf{W}_V^{(l,i)} = (\mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{I})^{\mathrm{T}}$ so that $(\mathbf{h}_i^{(l)})^{\mathrm{T}}\mathbf{W}_V^{(l,i)} = \mathbf{p}_i$.

$$\mathbf{b}_e = \sum_{a=i}^{i-1} (\mathbf{h}_i^{\mathrm{T}}\mathbf{W}_Q \cdot \mathbf{h}_a^{\mathrm{T}}\mathbf{W}_K)\mathbf{h}_i^{\mathrm{T}}\mathbf{W}_V = \sum \mathbf{p}_a \mathbf{1}(v_a^e = v_i^e) \tag{6}$$

base $= (2^0, 2^1, \cdots, 2^L)^{\mathrm{T}}$ and $\mathbf{u} = (\{\text{base} \cdot \mathbf{b}_{e_j}\}_{j=1}^{|E|})$. Obvious, there is $\mathbf{W}_O$ such that $\sum \mathbf{b}_e \mathbf{W}_O = (\mathbf{0}, \mathbf{0}, \mathbf{u}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0})$.

For embedding of y, using the copy past of Lemma C.1, we can obtain $\mathbf{h}_{2i} = (\mathbf{0}, \mathbf{l}_i, \mathbf{u}_{2i-1}, \mathbf{u}_{2i}, \mathbf{0}, \mathbf{p}_i)$. According to Lemma C.2, there exists $W_1^{(l)}, W_2^{(l)}$, such that we have $\mathbf{h}_{2i} = (\mathbf{0}, \mathbf{l}_i, \mathbf{u}_{2i-1}, \mathbf{u}_{2i}, \mathbf{m}_{2i}, \mathbf{p}_i)$, where $\mathbf{m}_{2i} = \mathrm{Relu}(\mathbf{u}_{2i-1} - \mathbf{u}_{2i}) + \mathrm{Relu}(\mathbf{u}_{2i} - \mathbf{u}_{2i-1})$.

Layer2

In this layer, for y token, we apply Lemma C.1 to copy $\mathbf{m}_{2i-2}$ from $\mathbf{h}_{2i-2}$ to $\mathbf{h}_{2i}$. Therefore, we have $\mathbf{h}_{2i} = (\mathbf{0}, \mathbf{l}_i, \mathbf{u}_{2i-1}, \mathbf{u}_{2i}, \mathbf{m}_{2i} + \mathbf{m}_{2i-2}, \mathbf{p}_i)$

For each x, copy corresponding $\mathbf{m}$ from its previous y and yielding the result that $\mathbf{h}_{2i-1} = (\mathbf{f}_i, \mathbf{0}, \mathbf{u}_{2i-1}, \mathbf{0}, \mathbf{m}_{2i-2}, \mathbf{p}_i)$.

In MLP, we calculate $\mathbf{f}_i = \mathrm{Relu}(\mathbf{h}_{2i-1}^{\mathrm{T}} \mathbf{W}_1^{(l)'}) = \mathrm{Relu}([\{\mathbf{f}_{2i-1}^{(e_j)} - M\mathbf{m}_{2i-2}[j]\}_{j=1}^{|E|}, \mathbf{0}])$.

Layer3

Setting $W_Q = W_K = (\mathbf{I}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0})$, we have $\mathbf{h}_i^{\mathrm{T}} \mathbf{W}_Q = \mathbf{h}_i^{\mathrm{T}} \mathbf{W}_K = \mathbf{f}_i^y$. Setting $\mathbf{W}_V$ such that $\mathbf{h}_i^{\mathrm{T}} \mathbf{W}_V = \mathbf{l}_i$.

According to Lemma C.2, there exists MLP such that we transfer the l^y to the output form, which served as the prediction.

Performance analysis

Here, we will analyze the sequence representation score and the in-context learning accuracy of our constructed model.

Sequence representation score

The probability for a example having same value of a factor as query sample is $\frac{1}{n_v}$ and the probability for having different values is $\frac{n_v - 1}{n_v}$. Therefore, the probability for the in-context examples have j samples have a same value of a factor is $\binom{i}{j} \frac{(n_v-1)^{i-j}}{n_v^i}$. We cannot distinguish k factors to decide which one is the hidden factor if the k factors shared a similar pattern matching results. The probility for this is that

$$\binom{i}{j}\binom{|E|}{k} \left(\frac{(n_v-1)^{i-j}}{n_v^i}\right)^k \left(1 - \frac{(n_v-1)^{i-j}}{n_v^i}\right)^{|E|-k}.$$

When we cannot distinguish the hidden factor from k factors, the probability to predict wrong results is $\frac{k-1}{k}$. Combining the results above, we obtain the error that

$$\sum_{j=0}^{i} \binom{i}{j} \sum_{k=2}^{|E|} \binom{|E|}{k} \frac{k-1}{k} \left(\frac{(n_v-1)^{i-j}}{n_v^i}\right)^k \left(1 - \frac{(n_v-1)^{i-j}}{n_v^i}\right)^{|E|-k}.$$

The probability to give a right prediction is

$$s_i = 1 - \sum_{j=0}^{i} \binom{i}{j} \sum_{k=2}^{|E|} \binom{|E|}{k} \frac{k-1}{k} \left(\frac{(v-1)^{i-j}}{v^i}\right)^k \left(1 - \frac{(v-1)^{i-j}}{v^i}\right)^{|E|-k}.$$

If we autogressing predict the hidden factor value and combining the results of the previous prediction, we have:

$$\mathrm{srs}_i = (1 - \mathrm{srs}_{i-1}) s_i + \mathrm{srs}_{i-1},$$

where $\mathrm{srs}_0 = s_0$.

In-context learning score

The copy-past mechanism is used to predict the answer of the query example. For the copy-past mechanism, have a in-context example with same prediction result as the query example is neccesary. When we correct predict the hidden factor, the probility to predict correctly is $1 - (\frac{v-1}{v})^i$. When we predict a wrong hidden factor, the probility is $\frac{1}{v}$. Combine the two above, we obtain the accuracy

$$(1 - (\frac{v-1}{v})^i)\,\mathrm{srs}_i + \frac{1}{v}(1 - \mathrm{srs}_i).$$

Because when no in-context example is given, the accuracy is $\frac{1}{v}$. Therefore we obtain the in-context learning score

$$\mathrm{cls}_i = \frac{(v-1)(v^{i-1} - (v-1)^{i-1})}{v^i}\,\mathrm{srs}_i\,.$$

□

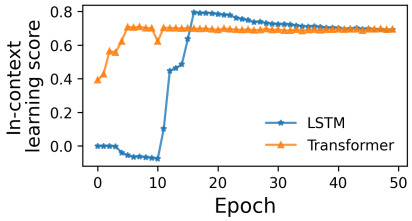

Figure 8: Comparison between the LSTM(Hochreiter & Schmidhuber, 1997) and Transformer.

D COMPARISION BETWEEN LSTM AND TRANSFORMER

In Section 3, our primary focus is on discussing the results of the Transformer. In this section, we aim to compare the in-context learning abilities of the Transformer and LSTM models. The LSTM model consists of 6 layers with 768 hidden dimensions, while the Transformer has 4 heads, 6 layers, and 64 hidden dimensions. We examine the scenario where $D_{\text{fix}} \to D_{\text{fix}}$. Unfortunately, we are unable to train the LSTM model for all other cases. During the comparison, we employ a larger hidden size for the LSTM model, as it tends to fail when using a smaller hidden size.

Our conclusion is

- LSTM is much harder for training than Transformer.
- LSTM has potential to obtain better in-context learning ability.