Do Large Language Models Have Compositional Ability? An Investigation into Limitations and Scalability

Zhuoyan Xu[*], **Zhenmei Shi**[*], **Yingyu Liang**
University of Wisconsin-Madison
{zxu444,zhmeishi,yliang}@cs.wisc.edu

Abstract

Large language models (LLM) have emerged as a powerful tool exhibiting remarkable in-context learning (ICL) capabilities. In this study, we delve into the ICL capabilities of LLMs on composite tasks, with only simple tasks as in-context examples. We develop a test suite of composite tasks that include logical and linguistic challenges and perform empirical studies across different LLM families. We observe that models exhibit divergent behaviors: (1) For simpler composite tasks that contains different input segments, the models demonstrate decent compositional ability, while scaling up the model enhances this ability; (2) for more complex composite tasks that involving sequentiaL reasoning, models typically underperform, and scaling up provide no improvements. We offer theoretical analysis in a simplified setting. We believe our work sheds new light on the capabilities of LLMs in solving composite tasks regarding the nature of the tasks and model scale. Our dataset and code is available at https://github.com/OliverXUZY/LLM_Compose.

1 Introduction

Large language models (LLM) have revolutionized general AI community. In this paper, we focus on the problem of how LLMs tackle composite tasks that incorporate multiple simple tasks. Specifically, we investigate whether a model trained/in-context learned on individual tasks can effectively integrate these skills to tackle combined challenges, which are intuitive and simple for humans. For instance, in Figure 1, if a human is given examples where words following an asterisk (*) will be capitalized and words surrounded by parenthesis will be permuted, one can also know words following an asterisk (*) surrounded by parenthesis will be capitalized and permuted simultaneously. This basic generalization seems trivial, yet we observe LLMs fail to generalize in this way.

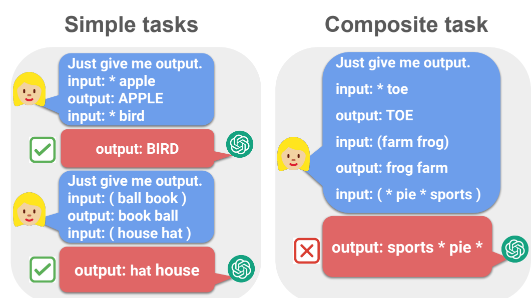

Figure 1: Inconsistent performance in GPT-4. Consider 2 simple tasks: If a word is followed by (*), capitalize the letter. If two words are surrounded by parenthesis, swap the positions. GPT-4 solves two simple tasks based on demonstrations (left). The composite tasks have test input with both asterisk (*) and parenthesis. The correct answer should be *output: SPORTS PIE*. However, GPT-4 fails to solve composite tasks (right). The same failure was observed in Claude 3.

Inspired by this observation, we further evaluate LLMs on a series of compositional tasks through ICL. The models were presented with examples of simple tasks and then asked to tackle composite tasks that they had not encountered during pretraining or in-context learning. We observe various behaviors: (1) for some composite tasks, the models showed a reasonable level of compositional skill, a capability that improved with larger model sizes; (2) for more complex composite tasks requiring sequential reasoning, the model struggle, and increasing the model size typically did not lead to better performance. Our key intuition is if the simple tasks forming a composite task can be easily separated into sub-tasks based on the inputs (e.g.,

[*]Equal contribution

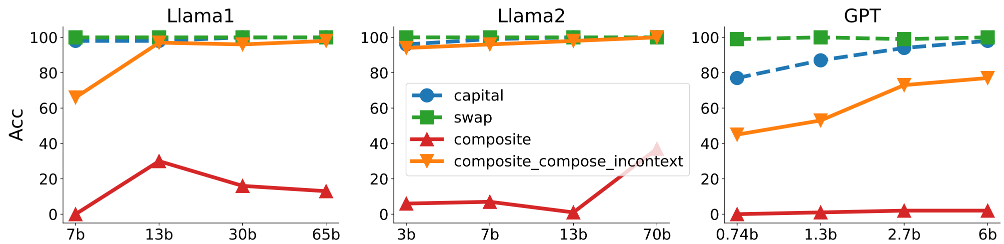

Figure 2: The exact match accuracy (y-axis) vs the model scale (x-axis, "b" stands for billion) for **Capitalization & Swap** (example in Figure 1). Line *capital*: performance on the simple task of capitalization; *swap*: on the simple task of swap; *composite*: in-context examples are from simple tasks while test input from the composite task. *composite incontext*: in-context examples and test input are all from the composite task (example in Table 3).

performed separately on different parts of the input sentence), the model is more likely to successfully complete such composite task (we call it "separable composite task").

Our contributions include both empirical experiments and theoretical analysis. We introduce a variety of composite tasks from both linguistic and logical domains to explore how the nature of these tasks influences the compositional performance of LLMs through ICL in experiments. Then we provide theoretical analysis on a simple yet insightful model: a one-layer single-head linear self-attention network. We demonstrate a clear separation in input embedding effectively breaking down composite tasks into simpler components. Due to the constraints of the page limit, we refer readers to Appendix A for details for related work.

2 Composite Logical Tasks

Our goal is to understand the LLMs' behavior on compositional reasoning tasks. We consider the standard in-context learning setting which concatenates $K = 10$ input-output examples and one testing input as the prompt for the LLM. We perform experiments across various LLM families, e.g., Llama families (Touvron et al., 2023) and GPTs (Radford et al., 2019; Black et al., 2021), see model details in Appendix B.2.

Warm-up setting. As a warm-up, we evaluate the **Capitalization & Swap** tasks (Figure 1) on different models. To make thorough evaluations, we consider four settings: (1) capital: only on the capitalization task; (2) swap: only on swap; (3) composite: in-context examples are from simple tasks while the test input is about the composite task; (4) composite in-context: in-context examples and the test input are all drawn from the composite task. The composite in-context setting reduces the evaluation to another simple task, not requiring the model to composite the simple task ability but directly learning from the in-context examples. It serves as the gold standard performance for the composite task. See Table 3 in Appendix B.1 for illustration.

Results. In Figure 2, somewhat surprisingly, we observe that LLMs cannot solve the composite task although they perform well on simple tasks. There is a significant gap between the performance in these settings. Models in Llama families can solve capital and swap with nearly $\sim$90% accuracy, but only achieve around 20% or below on the composite task. We also observe that composite in-context examples will significantly improve the performance: The accuracy of Llama families can go up to match the simple task accuracy. These observations show that *the models fail to compose the knowledge from the simple tasks, although they do have the representation power to solve the composite task* (which can only be exploited when provided composite in-context examples and scaling up does not help).

The experiment on Capitalization & Swap shows failure cases while existing studies reported some successful composite abilities (Levy et al., 2022; An et al., 2023b).

Logical tasks suite. We enhance our suite of logical tasks by introducing a series of straightforward tasks that process either simple words or numerical values, with the output being a specific functional transformation of the input. These tasks are detailed in Table 1.

Composite tasks are created by merging two simple tasks. We conceptualize simple tasks as functions, $f(\cdot)$ and $g(\cdot)$ that map inputs to their respective outputs. We identify two distinct approaches to creating composite tasks: **(1) Compose by parts**: For inputs x, y, the result is $f(x), g(y)$. **(2)**

Compose by steps: Given input x, the result is $f(g(x))$, such as **(A) + (B)** in Figure 1. We use customized symbol as function mapping for composing two simple tasks. We refer detailed illustration of how we compose tasks together and compose by parts or steps to Appendix B.1.

Tasks	Task	Input	Output
Words	(A) Capitalization	apple	APPLE
	(B) Swap	bell ford	ford bell
	(C) Two Sum	twenty @ eleven	thirty-one
	(D) Past Tense	pay	paid
	(E) Opposite	Above	Below
Numerical	(F) Plus One	435	436
	(G) Modular	15 @ 6	3
	(H) Two Sum Plus One	12 # 5	18

Table 1: This table contains a collection of simple logical tasks. The *Words* category encompasses tasks that modify words at the character or structural level. In contrast, the *Numerical* category is devoted to tasks that involve arithmetic computations performed on numbers.

Results. We provide our main results on composite tasks in Table 2. For the compose by parts tasks **(A) + (F)** and **(D) + (F)**, the models show strong compositional ability: the composite accuracy is high, improves with increasing scale, and eventually reaches similar performance as the "gold standard" composite in-context setting, as highlighted in red numbers. We refer these tasks as "separable composite tasks" which are relatively easy for model to solve. On the compose by steps tasks, we observe the models have various performance. For composite tasks with sequential reasoning steps, the models exhibit various performance. For tasks involving capitalization **(A)** or swap **(B)**, the model has poor performance in small scale (7b or lower) but have increased performance in increased model scale, such as 44% accuracy in **(A) + (C)** and 66% accuracy in **(B) + (D)**. On composite steps tasks involving arithmetic calculation of numerical numbers **(G) + (H)** the model has the worst performance and increasing model scale does not provide benefits. A key observation is that compose by part tasks are separable compositions, where the input can be broken down into two distinct segments. Such tasks are typically straightforward for a model to address. In all experiments, providing composed examples as in-context demonstration will help the model understand the composite tasks and solve them well, such as **Com. in-context** rows in all task combinations. We conclude models fail to compose mechanisms of two simple tasks together, however, given composite examples, models can learn the composed mechanism efficiently. More experimental details and results can be found in Appendix B.

Due to page limit, we refer readers for Composite Linguistic Translation Task to Appendix C.

Discussion. We observe the capability of models to handle composite tasks is significantly influenced by the task characteristics. Especially, if composite tasks contain simple tasks related to different parts or perspectives of the input, the model will tackle the composite tasks well. One natural explanation is the model processes the input in some hidden embedding space, and decomposes the embedding of the input into different "regions". Here each region is dedicated to specific types of information and thus related to different tasks — such as word-level modifications, arithmetic calculations, mapping mechanisms, semantic categorization, linguistic acceptability, or sentiment analysis. Then if the two simple tasks correspond to two different task types where they relates to separate regions of the embedding, the model can effectively manage the composite task by addressing each simple task operation within its corresponding region. As the model scale increases, its ability to handle individual tasks improves, leading to enhanced performance on composite tasks in such scenarios. For separable composite tasks, the inputs are divided into distinct regions and also reflect in embeddings, which results in high performance from the model. However, when the simple tasks are not separable (e.g., requiring sequential steps in reasoning), their information mixes together, complicating the model's ability to discern and process them distinctly. Such overlap often leads to the model's inability to solve the composite task. Such intuition is formalized in the following sections in a stylized theoretical setting.

3 Theoretical Analysis

Despite the complex nature of non-linearity in transformers in LLMs, we note it is useful to appeal to the simple case of linear models to see if there are parallel insights that can help us better understand the phenomenon. In this section, we provide an analysis of a linear attention module. We aim to provide rigorous proof about why LLMs can achieves compositional ability in some simple cases that could shed light on the more intricate behaviors observed in LLMs.

Pretrained	Tasks	Mistral 7B	Mistral 8x7B	Llama2 7B	Llama2 13B	Llama2 70B	Llama1 7B	Llama1 13B	Llama1 30B	Llama1 65B
(A) + (B)	Capitalization	99	98	99	100	100	98	98	100	100
	swap	100	100	100	100	100	100	100	100	100
	Compose	16	42	7	1	37	0	30	16	13
	Com. in-context	95	96	96	98	100	66	97	96	98
(A) + (C)	twoSum	71	100	72	93	99	62	56	98	99
	Capitalization	98	99	100	95	99	97	98	99	99
	Compose	8	19	3	23	44	3	3	31	2
	Com. in-context	31	65	52	77	100	9	22	93	69
(A) + (F)	Capitalization	97	99	98	77	99	84	96	99	98
	PlusOne	100	99	100	100	100	100	100	100	100
	Compose	92	96	74	69	97	57	60	69	99
	Com. in-context	99	98	99	100	100	99	99	100	100
(B) + (D)	Swap	100	100	100	100	100	100	100	100	100
	Past Tense	97	99	97	100	99	97	98	100	100
	Compose	6	12	0	1	62	57	34	46	5
	Com. in-context	92	98	86	95	98	86	95	89	94
(B) + (E)	Swap	100	100	100	100	100	100	100	100	100
	Opposite	61	62	58	68	65	51	58	64	63
	Compose	0	0	0	0	0	0	0	0	0
	Com. in-context	35	32	12	37	37	0	9	7	9
(D) + (F)	Past Tense	100	100	98	100	100	100	100	100	100
	Plus One	100	100	100	100	100	99	100	100	100
	Compose	71	46	32	80	80	40	44	14	74
	Com. in-context	98	100	98	99	100	95	96	98	100
(G) + (H)	Modular	25	22	5	23	43	9	16	29	29
	twoSumPlus	38	42	3	77	90	14	10	40	87
	Compose	4	5	0	1	1	0	0	0	5
	Com. in-context	4	8	13	13	12	11	13	7	12

Table 2: Results evaluating composite tasks on various models. The accuracy are showed in %.

Theoretical setup. We follow existing work (Akyürek et al., 2023; Garg et al., 2022; Mahankali et al., 2023) with slight generalization to K simple tasks. A labeled example is denoted as (x, y) where $x \in \mathbb{R}^d, y \in \mathbb{R}^K$. In a simple task $k \in [K]$, y has only one non-zero entry $y^{(k)}$. In a composite task, y can have non-zero entries in dimensions corresponding to the combined simple tasks. Following the previous work (Zhang et al., 2023b; Garg et al., 2022; Mahankali et al., 2023), we formulate pretraining on a linear attention as a linear regression problem.

We now detail how to evaluate the model on downstream *composite* tasks. We consider the downstream classification task to be a multi-class classification problem, where the output label is a K-dimensional vector and each entry corresponds to a simple task of binary classification. For any given simple task k, the binary classification label is given by sgn$(y_q^{(k)})$, where sgn is the sign function. Similarly, our prediction is $\widetilde{y}_q^{(k)} = \text{sgn}\left(\widehat{y}_q^{(k)}\right)$. The accuracy of a composite task is defined as $\text{Acc}_\theta(x_1, \ldots, y_N, x_q) = \frac{1}{K}\sum_{k=1}^{K} \mathbb{1}\left(\text{sgn}\left(\widehat{y}_q^{(k)}\right) = \text{sgn}(y_q^{(k)})\right)$. When x_q clear from context, we denote it as $\text{Acc}_\theta(\{x_i, y_i\}_{i=1}^N)$. We assume $x \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \Lambda)$, where $\Lambda \in \mathbb{R}^{d\times d}$ is the covariance matrix. Assume $y = Wx$, where $W \in \mathbb{R}^{K\times d}$. Then for any simple task $k \in [K]$, its label is the k-th entry of y, which is $y^{(k)} = \langle w^{(k)}, x \rangle$, where $w^{(k)}$ is the k-th row of W. We also assume each task weight $w^{(k)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_d)$. We refer readers see detailed setup in Appendix D.

Theoretical Analysis Results. In this section, we present our theoretical results. We explain the observation in empirical results through the lens of confined supports in input embeddings corresponding to separate subspaces (modeling separable composition). We set up our framerork as *Disjoint subspaces of simple tasks.* Recall that x lies in a d-dimensional space where each dimension represents a different characteristic. A simple task may depend only on a subset of these dimensions since its label only depends on a few features. Let $\mathbb{S} = [d]$ represent the dimensions of x. For a task k, the output $y^{(k)} = \langle w_k, x \rangle$ depends on a subset of dimensions in x. Denote this subset by $\mathbb{K} \subseteq \mathbb{S}$ and call it the active index set for task k. In practice, the dimensions within $\mathbb{K}$ could be associated with arithmetic operations, while those in $\mathbb{G}$ might pertain to semantic analysis. This illustrates the model's approach to segregate and address these tasks in their respective subspaces. We now introduce a mild assumption regarding the distribution of input embeddings.

Assumption 1. *Given two disjoint subspaces $\mathbb{K}$ and $\mathbb{G}$, the covariance matrix Λ of the input distribution can be segmented into block matrices $\Lambda_{\mathbb{K}\mathbb{K}}, \Lambda_{\mathbb{K}\mathbb{G}}, \Lambda_{\mathbb{G}\mathbb{K}}$, and $\Lambda_{\mathbb{G}\mathbb{G}}$, then we assume $\sigma_{max}(\Lambda_{\mathbb{K}\mathbb{G}}) = \sigma_{max}(\Lambda_{\mathbb{G}\mathbb{K}}) \leq \epsilon$ for constant ϵ, where $\sigma(\cdot)$ denote the singular value of matrix.*

Assumption 1 implies that for two separate simple tasks, each associated with its respective feature subspace $\mathbb{K}$ and $\mathbb{G}$, the covariance between these two sets of features is zero. This is a natural assumption. Suppose we have input embeddings from two distinct tasks, such as sentiment analysis and arithmetic computations. This assumption suggests that the feature subspaces of the input embeddings for these tasks are independent.

Consider a composite task $\mathcal{T}$ that combines two simple tasks k and g. Let $\mathcal{S}_k$ denote N labeled examples from task k, and similarly for $\mathcal{S}_g$. Given an x_q from composite task $\mathcal{T}$, we then define the model has **compositional ability** on $\mathcal{T}$ using $\mathcal{S}_{k\cup g}$ if the model has higher accuracy using these in-context examples, i.e. $\max\{\text{Acc}_\theta(\mathcal{S}_k), \text{Acc}_\theta(\mathcal{S}_g)\} \leq \text{Acc}_\theta(\mathcal{S}_k \cup \mathcal{S}_g)$. With the above definition, we will then explain the observed model behavior in empirical results, in particular, when distinct simple tasks have confined supports in input embeddings modeling the separable composition. We now define confined support, which means the input embedding of each task only has support within each task's feature subspace.

Definition 1 (Confined Support)**.** *We say a task has confined support if the input x only has larger singular values within its active index set. The norm of entries outside active index set be bounded by a small constant δ.*

This definition shows that each simple task only has large values within its corresponding subsets of dimensions of input embeddings. For example, let $\mathbb{K}$ represent the first d_1 dimensions of an input vector x, and $\mathbb{G}$ account for the remaining d_2 dimensions, with the total dimension being $d = d_1 + d_2$. The examples from task k will have input as $x = (x_1, x_{\delta_1})$ where $x_1 \in \mathbb{R}^{d_1}, x_{\delta_1} \in \mathbb{R}^{d_2}, \|x_{\delta_1}\| \leq \delta$. Similarly, the examples from task g will have inputs as $x = (x_{\delta_2}, x_2)$.

We now present our results of the compositional ability under a confined support of x.

Theorem 1. *Consider distinct tasks k and g with corresponding examples $\mathcal{S}_k, \mathcal{S}_g$, If two tasks have confined support, assume Assumption 1, with high probability, the model has the compositional ability. Moreover,*

$$Acc_\theta(\mathcal{S}_k) + Acc_\theta(\mathcal{S}_g) \leq Acc_\theta(\mathcal{S}_{k\cup g}).$$

Theorem 1 shows the compositional ability of LLMs to handle composite tasks that integrate two simple tasks, which have confined support in their own feature subspace.

An illustrative case involves the tasks of Capitalization **(A)** & Plus One **(F)** and Past Tense **(D)** & Plus One **(F)**, as depicted in Table 2. These two simple tasks involve word-level modification and arithmetic operation on separate parts of the input. Due to this separation, each task correlates with a specific segment of the input embedding. Therefore, it is observed that these tasks possess confined supports.

We provide additional theoretical results in Appendix E. We further provide Corollary 1 in Appendix E.1 illustrating the **necessity of the confined supports**, demonstrating that a model's failure to solve tasks with mixed steps reasoning, where contains overlapping input embedding spaces, thereby diminishing the model's ability to solve them when presented together. We also show the **scaling effect**: if simple tasks have confined support, the compositional ability of language models will increase as the model scale increases in Theorem 2 in Appendix E.2. We demonstrate this by showing that the accuracy of the model on each simple task improves with a larger model scale. We finally provide a **case study** on confined support for illustration in Appendix E.3.

ACKNOWLEDGMENTS

The work is partially supported by Air Force Grant FA9550-18-1-0166, the National Science Foundation (NSF) Grants 2008559-IIS, CCF-2046710, and 2023239-DMS.

REFERENCES

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. In *The Eleventh International Conference on Learning Representations*, 2023.

Shengnan An, Zeqi Lin, Bei Chen, Qiang Fu, Nanning Zheng, and Jian-Guang Lou. Does deep learning learn to abstract? a systematic probing framework. In *The Eleventh International Conference on Learning Representations*, 2023a.

Shengnan An, Zeqi Lin, Qiang Fu, Bei Chen, Nanning Zheng, Jian-Guang Lou, and Dongmei Zhang. How do in-context examples affect compositional generalization? *arXiv preprint arXiv:2305.04835*, 2023b.

Sanjeev Arora and Anirudh Goyal. A theory for emergence of complex skills in language models. *arXiv preprint arXiv:2307.15936*, 2023.

Lukas Berglund, Meg Tong, Max Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. The reversal curse: Llms trained on" a is b" fail to learn" b is a". *arXiv preprint arXiv:2309.12288*, 2023.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow. Technical report, Zenodo, March 2021. If you use this software, please cite it using these metadata.

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*, 2020.

Yanda Chen, Ruiqi Zhong, Sheng Zha, George Karypis, and He He. Meta-learning via language model in-context tuning. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 719–730, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.53.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2019.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Xiang Ren, Allyson Ettinger, Zaid Harchaoui, and Yejin Choi. Faith and fate: Limits of transformers on compositionality. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple contrastive learning of sentence embeddings. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6894–6910, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.552.

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.

Xinyang Geng and Hao Liu. Openllama: An open reproduction of llama, May 2023. URL `https://github.com/openlm-research/open_llama`.

Jiuxiang Gu, Chenyang Li, Yingyu Liang, Zhenmei Shi, Zhao Song, and Tianyi Zhou. Fourier circuits in neural networks: Unlocking the potential of large language models in mathematical reasoning and modular arithmetic. *arXiv preprint arXiv:2402.09469*, 2024.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. Surface form competition: Why the highest probability answer isn't always right. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 7038–7051, 2021.

SU Hongjin, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, et al. Selective annotation makes language models better few-shot learners. In *The Eleventh International Conference on Learning Representations*, 2023.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5254–5276, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.319.

Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*, 2022.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Najoung Kim and Tal Linzen. COGS: A compositional generalization challenge based on semantic interpretation. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 9087–9105, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.731.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2021.

Itay Levy, Ben Bogin, and Jonathan Berant. Diverse demonstrations improve in-context compositional generalization. *arXiv preprint arXiv:2212.06800*, 2022.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, 2021.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. What makes good in-context examples for GPT-3? In Eneko Agirre, Marianna Apidianaki, and Ivan Vulić (eds.), *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pp. 100–114, Dublin, Ireland and Online, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.deelio-1.10.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8086–8098, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.556.

Arvind Mahankali, Tatsunori B Hashimoto, and Tengyu Ma. One step of gradient descent is provably the optimal in-context learner with one layer of linear self-attention. *arXiv preprint arXiv:2307.03576*, 2023.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. MetaICL: Learning to learn in context. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2791–2809, Seattle, United States, July 2022a. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.201.

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? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2022b.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. Cross-task generalization via natural language crowdsourcing instructions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 2022.

OpenAI. Introducing ChatGPT. `https://openai.com/blog/chatgpt`, 2022. Accessed: 2023-09-10.

OpenAI. GPT-4 technical report. *arXiv preprint arxiv:2303.08774*, 2023.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 2022.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. *OpenAI blog*, 2018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2655–2671, 2022.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*, 2022.

Zhenmei Shi, Jiefeng Chen, Kunyang Li, Jayaram Raghuram, Xi Wu, Yingyu Liang, and Somesh Jha. The trade-off between universality and label efficiency of representations from contrastive learning. In *International Conference on Learning Representations*, 2023a.

Zhenmei Shi, Junyi Wei, Zhuoyan Xu, and Yingyu Liang. Why larger language models do in-context learning differently? In *R0-FoMo:Robustness of Few-shot and Zero-shot Learning in Large Foundation Models*, 2023b.

Zhenmei Shi, Yifei Ming, Ying Fan, Frederic Sala, and Yingyu Liang. Domain generalization via nuclear norm regularization. In *Conference on Parsimony and Learning*, pp. 179–201. PMLR, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 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.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, Joao Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 35151–35174. PMLR, 23–29 Jul 2023.

Xinyi Wang, Wanrong Zhu, Michael Saxon, Mark Steyvers, and William Yang Wang. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a.

Zhen Wang, Rameswar Panda, Leonid Karlinsky, Rogerio Feris, Huan Sun, and Yoon Kim. Multitask prompt tuning enables parameter-efficient transfer learning. In *The Eleventh International Conference on Learning Representations*, 2023b.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022.

Jerry Wei, Le Hou, Andrew Kyle Lampinen, Xiangning Chen, Da Huang, Yi Tay, Xinyun Chen, Yifeng Lu, Denny Zhou, Tengyu Ma, and Quoc V Le. Symbol tuning improves in-context learning in language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023a.

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*, 2023b.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*, 2022.

Zhuoyan Xu, Zhenmei Shi, Junyi Wei, Yin Li, and Yingyu Liang. Improving foundation models for few-shot learning via multitask finetuning. In *ICLR 2023 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2023.

Zhuoyan Xu, Zhenmei Shi, Junyi Wei, Fangzhou Mu, Yin Li, and Yingyu Liang. Towards few-shot adaptation of foundation models via multitask finetuning. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=1jbh2e0b2K`.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Jiacheng Ye, Zhiyong Wu, Jiangtao Feng, Tao Yu, and Lingpeng Kong. Compositional exemplars for in-context learning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 39818–39833. PMLR, 23–29 Jul 2023.

Renrui Zhang, Jiaming Han, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, Peng Gao, and Yu Qiao. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*, 2023a.

Ruiqi Zhang, Spencer Frei, and Peter L Bartlett. Trained transformers learn linear models in-context. *arXiv preprint arXiv:2306.09927*, 2023b.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pp. 12697–12706. PMLR, 2021.

Appendix

In this appendix, we provide full related work in Appendix B.2. We provide full logical tasks settings and results in Appendix B and along with full settings and results for translation tasks in Appendix C. We provide full theoretical set up in Appendix D and full theoretical results in Appendix E. We provide full proof in Appendix F.

A Related Work

Large language model. LLMs are often Transformer-based (Vaswani et al., 2017) equipped with enormous size of parameters and pretrained on vast training data. Typical LLMs includes BERT Devlin et al. (2019), PaLM Chowdhery et al. (2022), LLaMATouvron et al. (2023), ChatGPT (OpenAI, 2022), GPT4 (OpenAI, 2023). Pretraining methods include masked language modeling (Devlin et al., 2019; Liu et al., 2019), contrastive learning (Gao et al., 2021; Shi et al., 2023a) and auto-regressive pretraining (Radford et al., 2018; 2019). Adapting LLMs to various downstream tasks has received significant attention, e.g., adaptor Hu et al. (2022; 2023); Zhang et al. (2023a); Shi et al. (2024), prompt tuning (Lester et al., 2021; Li & Liang, 2021; Wei et al., 2023a), multitask finetuning Sanh et al. (2022); Wang et al. (2023b); Xu et al. (2023; 2024), instruction tuning Chung et al. (2022); Mishra et al. (2022), in-context learning (Min et al., 2022b; Dong et al., 2022; Yao et al., 2023), reinforcement learning from human feedback (RLHF) Ouyang et al. (2022).

In-context learning. LLM exhibits a remarkable ability for in-context learning (ICL) (Brown et al., 2020), particularly for generative models. Given a sequence of labeled examples and a testing example (combined as a prompt), the model can construct new predictors for testing examples without further parameter updates. Several empirical works are investigating the behavior of ICLs. Zhao et al. (2021); Holtzman et al. (2021); Lu et al. (2022) formulate the problems and report the sensitivity. Rubin et al. (2022); Liu et al. (2022); Hongjin et al. (2023); Wang et al. (2023a) provide methods for better choosing in-context learning examples. Chen et al. (2022); Min et al. (2022a) use meta training with an explicit in-context learning object to boost performance. Theoretically, Xie et al. (2022); Garg et al. (2022) provide a framework to explain the in-context learning working mechanism. Von Oswald et al. (2023); Akyürek et al. (2023); Mahankali et al. (2023); Zhang et al. (2023b), investigating with linear models, show how transformers can represent gradient descent and conduct linear regression. Based on these works, we provide an analysis showing how LLM can exhibit compositional ability in ICL.

Emergence of compositional ability. Scaling law was first proposed by Kaplan et al. (2020) and then followed up by Hoffmann et al. (2022), emphasizing both on scale of models and training data. Sometimes, increasing scale can lead to new behaviors of LLMs, termed *emergent abilities* (Wei et al., 2022; Arora & Goyal, 2023; Gu et al., 2024). Recent works show LLMs with larger scales have distinct behavior compared to smaller language models (Wei et al., 2023b; Shi et al., 2023b). These behaviors can have positive or negative effects on performance. Solving complex tasks and reasoning is an active problem in the AI community Huang & Chang (2022). There is a line of empirical works investigating the compositional ability in linguistic fashion (Kim & Linzen, 2020; Levy et al., 2022; An et al., 2023a;b). LLMs are capable of learning abstract reasoning (e.g. grammar) to perform new tasks when finetuned or given suitable in-context examples. In our work, we include linguistic experiments as part of our testing suite, illustrating LLMs' compositional ability. Ye et al. (2023); Berglund et al. (2023); Dziri et al. (2023) show LLMs will have difficulties solving tasks that require reasoning. Berglund et al. (2023) studies that LLMs trained on "A is B" fail to learn "B is A". In our work, we conduct similar experiments showing LLMs will fail on composite if different steps of logical rules are mixed.

B Logical Tasks

We provide full explanation of logical composite tasks below. We first show compose in-context example in Table 3.

	Composite	**Composite in-context**
Prompt	*input: * apple* *output: APPLE* *input: (farm frog)* *output: frog farm* *input: * (bell ford)*	*input: (* good * zebra)* *output: ZEBRA GOOD* *input: (* model * math)* *output: MATH MODEL* *input: (* bicycle * add)*
Truth	*output: FORD BELL*	*output: ADD BICYCLE*

Table 3: Examples of two settings. Composite: in-context examples are about simple tasks while the test input is about the composite task. Composite in-context: both in-context examples and the test input are about the composite task.

Tasks	**Simple Task**	**Simple Task**	**Composite**
(A) + (B)	input: * apple output: APPLE	input: (farm frog) output: frog farm	input: (* bell * ford) output: FORD BELL
(A) + (F)	input: 435 output: 436	input: cow output: COW	input: 684 cat output: 685 CAT

Table 4: Examples of the two logical composite tasks. Full examples can be found in Appendix B.

B.1 Illustration of Logical Tasks

Composite tasks are created by merging two simple tasks. We conceptualize simple tasks as functions, $f(\cdot)$ and $g(\cdot)$ that map inputs to their respective outputs. We identify two distinct approaches to creating composite tasks: **(1) Compose by parts**: For inputs x, y, the result is $f(x), g(y)$. One example is **(A) + (F)** in Table 4 If numerical number is given, it will increment by one; if word is given, the letters will be capitalize; if both are given, perform both operations. **(2) Compose by steps**: Given input x, the result is $f(g(x))$. One example is **(A) + (B)** in Table 4. We use customized symbol as function mapping for composing two simple tasks. Examples can be found in Figure 1 and Table 4. Following existing work, we use exact match accuracy for evaluating the performance, since the output for these tasks is usually simple and short.

B.2 Full Logical Tasks

We provide full explanation of logical composite tasks below. Examples can be seen in Table 5.

- **(A) + (B) Capitalization & Swap**, as in Figure 1.
- **(A) + (C) Capitalization & Two Sum.** Given words of numerical numbers, ***** represents the operation of capitalizing, @ represents summing the two numbers.
- **(G) + (H) Modular & Two Sum Plus.** Given numerical numbers, **@** represents the operation of taking modular, **#** represents to sum the two numbers and then plus one.
- **(A) + (F) Capitalization & Plus One.** If numerical numbers are given, plus one; if words are given, capitalize the word; if both are given, perform both operations.

Among these, **(A) + (F)** performs the two operations on separable parts of the test inputs (i.e., separable composite task).

We design our logical tasks following the idea of math reasoning and logical rules. The details are shown in Table 5. Our numerical numbers in Table 1 are uniformly randomly chosen from 1 to 1000. The words of numbers in task **(C)** are uniformly randomly chosen from one to one hundred. The words representing objects in Table 1 are uniformly randomly chosen from class names of ImageNet, after dividing the phrase (if any) into words. We randomly choose 100 examples in composite testing data in our experiments and replicate the experiments in each setting three times. We fixed the number of in-context examples as $K = 10$ as demonstrations.

We use exact match accuracy for evaluating the performance between sequence output. The calculation of exact match accuracy divided the number of matched words by the length of ground truth.

Tasks	Simple Task	Simple Task	Composite
(A) + (B)	input: * apple output: APPLE	input: (farm frog) output: frog farm	input: (* bell * ford) output: FORD BELL
(A) + (C)	input: * *(five)* output: FIVE	input: *twenty* @ *eleven* output: thirty-one	input: * *(thirty-seven* @ *sixteen)* output: FIFTY-THREE
(G) + (H)	input: 15 @ 6 output: 3	input: 12 # 5 output: 18	input: 8 # 9 @ 7 Ouput: 4
(A) + (F)	input: 435 output: 436	input: cow output: COW	input: 684 cat output: 685 CAT

Table 5: Examples of the four logical composite tasks. Note that in **(G) + (H)**, the output of the composite task can be either 4 or 11 depending on the order of operations and we denote both as correct.

For Llama models, we use official Llama1 and Llama2 models from Meta (Touvron et al., 2023), we use open_llama_3b_v2 from open OpenLlama (Geng & Liu, 2023). For GPT models, we use GPT2-large from openAI (Radford et al., 2019), we use GPT-neo models for GPT models in other scales from EleutherAI (Black et al., 2021).

B.3 Results

We show visualization of some logical tasks accuracy along the increasing to model scale, complement to Table 2.

C Composite Linguistic Translation

C.1 Main Results

Inspired by previous works in compositional generalization (An et al., 2023b; Levy et al., 2022; An et al., 2023a; Kim & Linzen, 2020), here we design our composite tasks by formal language translation tasks.

Our translation tasks are mainly derived from semantic parsing task COGS (Kim & Linzen, 2020) and compositional generalization task COFE An et al. (2023b). These two datasets contain input as natural English sentences and output as a chain-ruled sentence following a customized grammar (see details in Appendix C.2). We construct two composite tasks centered on compositional generalization, utilizing the training datasets to create in-context examples. See details in Appendix C.2.

We use the word error rate (WER) as the metric. It measures the minimum number of editing operations (deletion, insertion, and substitution) required to transform one sentence into another, and is common for speech recognition or machine translation evaluations.

(T1) Phrase Recombination with Longer Chain. COFE proposed two compositional generalization tasks (Figure 2 in An et al. (2023b)). *Phrase Recombination*: integrate a prepositional phrase (e.g., “A in B”) into a specific grammatical role (e.g., “subject”, “object”); *Longer Chain*: Extend the tail of the logical form in sentences. We consider them as simple tasks, and merge them to form a composite task: substitute the sentence subject in the Longer Chain task with a prepositional phrase from the Phrase Recombination task. Details and examples are in Table 8 of Appendix C.2.

(T2) Passive to Active and Object to Subject Transformation. We consider two tasks from Kim & Linzen (2020). *Passive to Active*: Transitioning sentences from passive to active voice. *Object to Subject*: Changing the same object (a common noun) from objective to subjective. They are merged to form our composite task, where both transformations are applied simultaneously to the input sentence. Details and examples are in Table 7 of Appendix C.2.

Results. Figure 4 shows that LLMs are capable of handling these composite tasks. The WER on the composite task is a decent and improves with increasing model scale, particularly in Llamma2 models. These confirm the composite abilities of the models in these tasks.

Here we notice both composite tasks are separable composite tasks: if we break down these sentences into sub-sentences and phrases, the simple task operations occur in different parts or perspectives of

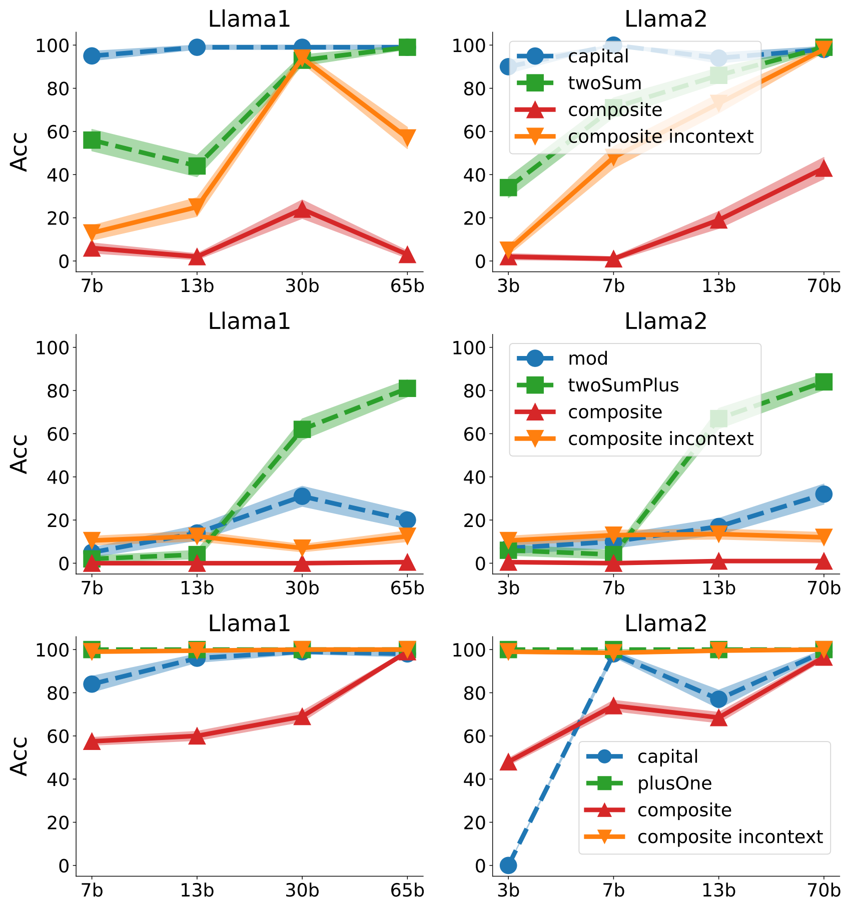

Figure 3: The accuracy v.s. model scale on composite logical rule tasks. Dashed lines: simple tasks. Solid lines: composite tasks. Rows: **(A)** + **(C)** Capitalization & Two Sum; **(G)** + **(H)** Modular & Two Sum Plus; **(A)** + **(F)** Capitalization & Plus One. Columns: different models. Lines: performance in different evaluation settings, i.e., the two simple tasks, the composite setting, and the composite in-context setting (examples for the last two are shown in Table 3).

the input sentences. So the results here provide further support for composite abilities on separable composite tasks where simple tasks forming the composite task are related to inputs in different parts or perspectives.

Discussion. We observe the capability of models to handle composite tasks is significantly influenced by the task characteristics. Especially, if composite tasks contain simple tasks related to different parts or perspectives of the input, the model will tackle the composite tasks well.

One natural explanation is the model processes the input in some hidden embedding space, and decomposes the embedding of the input into different "regions". Here each region is dedicated to specific types of information and thus related to different tasks — such as word-level modifications, arithmetic calculations, mapping mechanisms, semantic categorization, linguistic acceptability, or sentiment analysis. Then if the two simple tasks correspond to two different task types where they relates to separate regions of the embedding, the model can effectively manage the composite task by addressing each simple task operation within its corresponding region. As the model scale increases, its ability to handle individual tasks improves, leading to enhanced performance on composite tasks in such scenarios. For separable composite tasks, the inputs are divided into distinct regions and also reflect in embeddings, which results in high performance from the model. However, when the simple tasks are not separable (e.g., requiring sequential steps in reasoning), their information mixes together, complicating the model's ability to discern and process them distinctly. Such overlap often leads

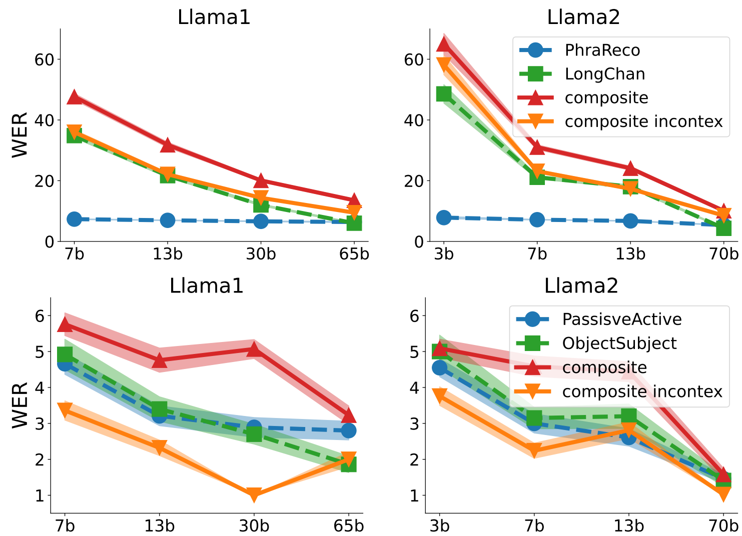

Figure 4: The word error rate (WER) vs the model scale on composite linguistic translation tasks. Dashed lines: simple tasks. Solid lines: composite tasks. Rows: **(T1)** Phrase Recombination with Longer Chain; **(T2)** Passive to Active and Object to Subject Transformation. Columns: different models. Lines: performance in different evaluation settings, e.g., the two simple tasks, the composite setting, and the composite in-context setting (examples are shown in Appendix C.2).

to the model's inability to solve the composite task. Such intuition is formalized in the following sections in a stylized theoretical setting.

C.2 Full Details

Our translation tasks mainly follow the compositional generalization tasks in COFE (An et al., 2023b). The details can be found in Section 4 in An et al. (2023a). We directly take the source grammar $\mathcal{G}_s$ in COGS which mimics the English natural language grammar, and reconstruct the target grammar $\mathcal{G}_t$ in COGS to be chain-structured.

We follow the Primitive coverage principle proposed by An et al. (2023b) that primitives contained in each test sample should be fully covered by in-context examples. Here, primitives refer to the basic, indivisible elements of expressions, including subjects, objects, and verbs. Note that multiple sets of in-context examples can meet these criteria for each test case. Across all experimental conditions, we maintain a consistent number of test instances at 800.

We use the word error rate (WER) as the metric. It measures the differences between 2 sentences. It measures the minimum number of editing operations (deletion, insertion, and substitution) required to transform one sentence into another, and is common for speech recognition or machine translation evaluations. The computation of WER is divided the nunber of operations by the length of ground truth.

Original Target Grammar	Chain-Structured Target Grammar
rose (x_1) AND help . theme (x_3 , x_1) AND help . agent (x_3 , x_6) AND dog (x_6)	HELP (DOG, ROSE, NONE)
* captain (x_1) ; eat . agent (x_2 , x_1)	EAT (CAPTION, NONE, NONE)
* dog (x_4) ; hope . agent (x_1 , Liam) AND hope . ccomp (x_1 , x_5) AND prefer . agent (x_5 , x_4)	HOPE (LIAM, NONE, NONE) CCOMP PREFER (DOG, NONE, NONE)

Table 6: Demonstration in An et al. (2023a) showing examples with the original grammar and the new chain-structured grammar.

In formal language tasks, as mentioned in Appendix C.1, we change the original target grammar of COGS to be chain-structured. In Table 6, we list some examples with the original target grammar and the new chain-structured grammar.

- First, to distinguish the input and output tokens, we capitalize all output tokens (e.g., from "rose" to "ROSE").
- Second, we replace the variables (e.g., "x_1") in the original grammar with their corresponding terminals (e.g., "ROSE").
- Then, we group the terminals of AGENT (e.g., "DOG"), THEME (e.g., "ROSE"), and RECIPIENT with their corresponding terminal of PREDICATE (e.g., "HELP") and combine this group of terminals in a function format, i.e., "PREDICATE (AGENT, THEME, RECIPIENT)". If the predicate is not equipped with an agent, theme, or recipient in the original grammar, the corresponding new non-terminals (i.e., AGENT, THEME, and RECIPIENT, respectively) in the function format above will be filled with the terminal NONE (e.g., "HELP (DOG, ROSE, NONE)"). Such a function format is the minimum unit of a CLAUSE.
- Finally, each CLAUSE is concatenated with another CLAUSE by the terminal CCOMP (e.g., "HOPE (LIAM, NONE, NONE) CCOMP PREFER (DOG, NONE, NONE)").

Task	In-context Example	Testing Example
Passive to Active	The book was squeezed . SQUEEZE (NONE , BOOK , NONE)	Sophia squeezed the donut . SQUEEZE (SOPHIA , DONUT , NONE)
Object to Subject	Henry liked a cockroach in a box . LIKE (HENRY , IN (COCKROACH , BOX)	A cockroach inflated a boy . INFLATE (COCKROACH , BOY , NONE)
Composite Task	The book was squeezed . SQUEEZE (NONE , BOOK , NONE) Henry liked a cockroach in a box . LIKE (HENRY , IN (COCKROACH , BOX)	A cockroach squeezed the hedgehog . SQUEEZE (COCKROACH , hedgehog , NONE)

Table 7: Testing examples of Passive to Active and Object to Subject, red text shows the verbs changing from passive to active voice in simple tasks, and blue text shows the nouns from objective to subjective.

Below we provide a detailed explanation of our two composite tasks in translation tasks.

Passive to Active and Object to Subject Transformation. Based on the generalization tasks identified in Kim & Linzen (2020)), we select two distinct challenges for our study as two simple tasks. *Passive to Active*: Transitioning sentences from Passive to Active voice. *Object to Subject*: Changing the focus from Object to Subject using common nouns. These tasks serve as the basis for our composite task, where both transformations are applied simultaneously to the same sentence. Examples illustrating this dual transformation can be found in Table 7.

Enhanced Phrase Subject with Longer Chain. COFE proposed two compositional generalization tasks (Figure 2 in An et al. (2023b)): *Phrase Recombination (PhraReco)*: integrate a prepositional phrase (e.g., "A in B") into a specific grammatical role (e.g., "subject", "object"); *Longer Chain (LongChain)*: Extend the tail of the logical form in sentences. We consider these two generalization tasks as two simple tasks, merging them to form a composite task. In particular, we substitute the sentence subject in the Longer Chain task with a prepositional phrase from the Phrase Recombination task, creating a more complex task structure. Detailed examples of this combined task can be found in Table 8.

Task		Example
Phrase Recombination	Input	The baby on a tray in the house screamed .
	Output	SCREAM (ON (BABY , IN (TRAY , HOUSE)) , NONE , NONE)
Longer Chain	Input	A girl valued that Samuel admired that a monkey liked that Luna liked that Oliver respected that Savannah hoped that a penguin noticed that Emma noticed that the lawyer noticed that a cake grew .
	Output	VALUE (GIRL , NONE , NONE) \ CCOMP ADMIRE (SAMUEL , NONE , NONE) \ CCOMP LIKE (MONKEY , NONE , NONE) \ CCOMP LIKE (LUNA , NONE , NONE) \ CCOMP RESPECT (OLIVER , NONE , NONE) \ CCOMP HOPE (SAVANNAH , NONE , NONE) \ CCOMP NOTICE (PENGUIN , NONE , NONE) \ CCOMP NOTICE (EMMA , NONE , NONE) \ CCOMP NOTICE (LAWYER , NONE , NONE) \ CCOMP GROW (NONE , CAKE , NONE)
Composite Task	Input	The baby on a tray in the house valued that Samuel admired that a monkey liked that Luna liked that Oliver respected that Savannah hoped that a penguin noticed that Emma noticed that the lawyer noticed that a cake grew .
	Output	VALUE (ON (BABY , IN (TRAY , HOUSE) , NONE , NONE) \ CCOMP ADMIRE (SAMUEL , NONE , NONE) \ CCOMP LIKE (MONKEY , NONE , NONE) \ CCOMP LIKE (LUNA , NONE , NONE) \ CCOMP RESPECT (OLIVER , NONE , NONE) \ CCOMP HOPE (SAVANNAH , NONE , NONE) \ CCOMP NOTICE (PENGUIN , NONE , NONE) \ CCOMP NOTICE (EMMA , NONE , NONE) \ CCOMP NOTICE (LAWYER , NONE , NONE) \ CCOMP GROW (NONE , CAKE , NONE)

Table 8: Testing examples of Phrase Recombination and Longer Chain, red text shows the phrase serving as primitives in sentences in simple tasks, and blue text shows the logical structures as sub-sentences in long sentences.

D Theoretical Analysis Setup: Separable in Embedding Space in the Linear Setting

Despite the complex nature of non-linearity in transformers in LLMs, we note that it is not necessarily easy to understand the source of behavior for linear models either. Indeed, it is useful to appeal to the simple case of linear models to see if there are parallel insights that can help us better understand the phenomenon. In this section, we provide an analysis of a linear attention module. We aim to uncover underlying principles that could shed light on the more intricate behaviors observed in LLMs.

In-context learning. We follow existing work (Akyürek et al., 2023; Garg et al., 2022; Mahankali et al., 2023) with slight generalization to K simple tasks. A labeled example is denoted as (x, y) where $x \in \mathbb{R}^d, y \in \mathbb{R}^K$. In a simple task $k \in [K]$, y has only one non-zero entry $y^{(k)}$. In a composite task, y can have non-zero entries in dimensions corresponding to the combined simple tasks. The model takes a prompt $(x_1, y_1, \ldots, x_N, y_N, x_q)$ as input, which contains N in-context examples (x_i, y_i)'s and a query x_q, and aims to predict $\hat{y}_q$ close to the true label y_q for x_q. The prompt is usually stacked into an embedding matrix:

$$E := \begin{pmatrix} x_1 & x_2 & \ldots & x_N & x_q \\ y_1 & y_2 & \ldots & y_N & 0 \end{pmatrix} \in \mathbb{R}^{d_e \times (N+1)}$$

where $d_e = d + K$. In in-context learning, we first pretrain the model using training prompts and then evaluate the model with evaluation prompts; see details below.

Pretraining procedure. We have B training data indexed by τ, each containing an input prompt $(x_{\tau,1}, y_{\tau,1}, \ldots, x_{\tau,N}, y_{\tau,N}, x_{\tau,q})$ and a corresponding true label $y_{\tau,q}$. Consider the following empirical loss:

$$\widehat{L}(\theta) = \sum_{k=1}^{K} \widehat{L}_k(\theta) = \frac{1}{2B} \sum_{\tau=1}^{B} \|\widehat{y}_{\tau,q} - y_{\tau,q}\|^2,$$

and the population loss (i.e., $B \to \infty$):

$$L(\theta) = \frac{1}{2} \mathbb{E}_{x_{\tau,1}, y_{\tau,1}, \cdots, x_{\tau,N}, y_{\tau,N}, x_{\tau,q}} \left[(\widehat{y}_{\tau,q} - y_{\tau,q})^2 \right].$$

Evaluation procedure. We now detail how to evaluate the model on downstream *composite* tasks. We consider the downstream classification task to be a multi-class classification problem, where the output label is a K-dimensional vector and each entry corresponds to a simple task of binary classification. For any given simple task k, the binary classification label is given by $\text{sgn}(y_q^{(k)})$, where sgn is the sign function. Similarly, our prediction is $\widetilde{y}_q^{(k)} = \text{sgn}\left(\widehat{y}_q^{(k)}\right)$. The accuracy of a composite task is defined as

$$\text{Acc}_\theta(x_1, \ldots, y_N, x_q) = \frac{1}{K} \sum_{k=1}^{K} \mathbb{1}\left(\text{sgn}\left(\widehat{y}_q^{(k)}\right) = \text{sgn}(y_q^{(k)})\right).$$

When x_q clear from context, we denote it as $\text{Acc}_\theta(\{x_i, y_i\}_{i=1}^{N})$.

Data. Assume $x \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \Lambda)$, where $\Lambda \in \mathbb{R}^{d\times d}$ is the covariance matrix. Assume $y = Wx$, where $W \in \mathbb{R}^{K\times d}$. Then for any simple task $k \in [K]$, its label is the k-th entry of y, which is $y^{(k)} = \left\langle w^{(k)}, x \right\rangle$, where $w^{(k)}$ is the k-th row of W. We also assume each task weight $w^{(k)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_d)$.

Linear self-attention networks. These networks are widely studied (Von Oswald et al., 2023; Akyürek et al., 2023; Mahankali et al., 2023; Garg et al., 2022; Zhang et al., 2023b; Shi et al., 2023b). Following them, we consider the following linear self-attention network with parameters $\theta = (W^{PV}, W^{KQ})$:

$$f_{\text{LSA},\theta}(E) = E + W^{PV} E \cdot \frac{E^\top W^{KQ} E}{N}.$$

The prediction of the model for x_q is $\hat{y}_q = [f_{\text{LSA},\theta}(E)]_{(d+1):(d+K),N+1}$, the bottom rightmost sub-vector of $f_{\text{LSA},\theta}(E)$ with length K. Let

$$W^{PV} = \begin{pmatrix} W_{11}^{PV} & W_{12}^{PV} \\ (W_{21}^{PV})^\top & W_{22}^{PV} \end{pmatrix} \in \mathbb{R}^{(d+K)\times(d+K)}$$
$$W^{KQ} = \begin{pmatrix} W_{11}^{KQ} & W_{12}^{KQ} \\ (W_{21}^{KQ})^\top & W_{22}^{KQ} \end{pmatrix} \in \mathbb{R}^{(d+K)\times(d+K)},$$

where $W_{11}^{PV} \in \mathbb{R}^{d\times d}$, $W_{12}^{PV}, W_{21}^{PV} \in \mathbb{R}^{d\times K}$, and $W_{22}^{PV} \in \mathbb{R}^{K\times K}$; similar for W^{KQ}. Then the prediction is

$$\widehat{y}_q = \begin{pmatrix} (W_{21}^{PV})^\top & W_{22}^{PV} \end{pmatrix} \left(\frac{EE^\top}{N} \right) \begin{pmatrix} W_{11}^{KQ} \\ (W_{21}^{KQ})^\top \end{pmatrix} x_q. \tag{1}$$

We observe only part of the parameters affect our prediction, so we treat the rest of them as zero in our analysis.

E Theory for Confined Support

E.1 Necessity of the Confined Supports.

In this section we demonstrate that when the confined support is violated, the simple tasks begin to exhibit variations (large singal values) across the entire feature subspace of the input embedding.

For instance, the composite task of Capitalization (A) & Swap (B), which involves mixed steps in reasoning as shown in Figure 2, shows poor performance of LLMs given both simple tasks' examples as in-context demonstrations. Another example is Modular (G) & Two Sum Plus (H) as shown in the last row of Table 2, where both simple tasks involve multisteps arithmetic operation. These two tasks share the same support on embedding space, mixing their variations and leading to the model's inability to effectively address the composite tasks that integrate them. Below we will provide a theorem establishing that if two tasks share overlapping support in the embedding space, there can be a scenario where the model fails to exhibit compositional ability.

Corollary 1. *If two tasks do not have confined support, there exists one setting which we have*

$$Acc_\theta(\mathcal{S}_k) = Acc_\theta(\mathcal{S}_g) = Acc_\theta(\mathcal{S}_{k\cup g}).$$

Corollary 1 demonstrates that a model's failure to solve tasks with mixed steps reasoning, where contains overlapping input embedding spaces, thereby diminishing the model's ability to solve them when presented together.

E.2 Compositional Ability with Model Scale

We then show if simple tasks have confined support, the compositional ability of language models will increase as the model scale increases. We demonstrate this by showing that the accuracy of the model on each simple task improves with a larger model scale.

Note that the optimal solutions of parameter matrices as W^{*PV} and W^{*KQ}. We naturally consider that the rank of the parameter matrices W^{*PV} and W^{*KQ} can be seen as a measure of the model's scale. A higher rank in these matrices implies that the model can process and store more information, thereby enhancing its capability. We state the theorem below:

Theorem 2. *Suppose a composite task satisfies confined support. Suppose we have $(x_1, y_1, \dots, x_N, y_N, x_q)$ as an testing input prompt, and corresponding W where $y_i = Wx_i$. As rank r decreases, $\mathbb{E}_{W,x_1,\cdots,x_N}[Acc_\theta]$ will have a smaller upper bound.*

Theorem 2 shows the expected accuracy of a model on composite tasks is subjected to a lower upper bound as the scale of the model diminishes. This conclusion explains why scaling-up helps the performance when the model exhibits compositional ability for certain tasks (those we called "separable composite task"). One common characteristic for these tasks is their inputs display confined supports within the embeddings. This is evidenced by the model's decent performance on tasks as presented in Table 2 and Figure 4, where inputs are composed by parts.

E.3 Case Study of Confined Support

Our theoretical conclusion shows the model behaviors regarding the input embedding. It states the model will have compositional ability if tasks are under confined support of input embedding. To illustrate such theoretical concepts and connect them to empirical observations, we specialize the general conclusion to settings that allow easy interpretation of disjoint. In this section, we provide a toy linear case study on classification tasks showing how confined support on embedding can be decomposed and composite tasks can be solved. We assume $\delta = \epsilon = 0$ in below simple example.

Consider there are only two simple tasks for some random objects with the color red and blue, and the shape square and round: (1) binary classification based on the color: red and blue. (2) binary classification based on shape: circle and square. However, during evaluation, the composite task is a four-class classification, including red circle, red square, blue circle, and blue square.

Then we have two simple tasks $K = 2$. Consider the input embedding $x = (a, b)$, where $a \in \mathbb{R}^2, b \in \mathbb{R}^2, d = 4$. Consider $W = \begin{pmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix}$ and $y = Wx$.

Consider the inputs from simple and composite tasks as:

- Task 1: Red: $x_1 = (1, 0, 0, 0), y_1 = (1, 0)$ and blue: $x_2 = (0, 1, 0, 0), y_2 = (-1, 0)$.
- Task 2: Circle $x_3 = (0, 0, 1, 0)\;\; y_3 = (0, 1)$ and square $x_4 = (0, 0, 0, 1)\;\; y_4 = (0, -1)$.

- Composed task: red circle $x_5 = (1, 0, 1, 0), y_5 = (1, 1)$, red square $x_6 = (1, 0, 0, 1), y_6 = (1, -1)$, blue circle $x_7 = (0, 1, 1, 0)$ $y_7 = (-1, 1)$ and blue square $x_8 = (0, 1, 0, 1)$ $y_8 = (-1, -1)$.

Suppose we have the optimal solution $\hat{y}_q$ as in Equation (1). Given $x_q = (1, 0, 1, 0)$ as a testing input of a red circle example. During the test, we have different predictions given different in-context examples:

1. Given only examples from Task 1 (red and blue): $[(x_1, y_1), (x_2, y_2)]$, we have $\widehat{y}_q = (1, 0)$ can only classify the color as red.
2. Given only examples from Task 2 (square and circle): $[(x_4, y_4), (x_3, y_3)]$, we have $\widehat{y}_q = (0, 1)$ only classify the shape as a circle.
3. Given a mixture of examples from Task 1 and 2 (red and circle): $[(x_1, y_1), (x_3, y_3)]$, we have $\widehat{y}_q = (1, 1)$ can classify as red and circle.

We can see that, in the final setting the model shows compositional ability. This gives a concrete example for the analysis in Theorem 1.

F Deferred Proof

In this section, we provide a formal setting and proof.

F.1 Proof of Compositional Ability under Confined Support

Here, we provide the proof of our main conclusion regarding Theorem 1 and Corollary 1.

Without abuse of notation, we denote $U = W_{11}^{KQ}, u = W_{22}^{PV}$.

We further add some mild assumptions.

1. The covariance matrix Λ of simple tasks will have the same trace, to prevent the scale effect of different simple tasks.
2. The spectral norm of Λ is bounded both sides $m \le \|\Lambda\| \le M$.

We first introduce the lemma where the language model only pretrained on one simple task ($K = 1$). The pretraining loss $L(\theta)$ can be re-factored and the the solution will have a closed form. We further

Lemma F.1 (Lemma 5.3 in Zhang et al. (2023b)). *Let* $\Gamma := \left(1 + \frac{1}{N}\right) \Lambda + \frac{1}{N} \operatorname{tr}(\Lambda) I_{d\times d} \in \mathbb{R}^{d\times d}$. *Let*

$$\tilde{\ell}(U, u) = \operatorname{tr}\left[\frac{1}{2} u^2 \Gamma\Lambda U\Lambda U^\top - u\Lambda^2 U^\top\right]$$

Then

$$\min_\theta L(\theta) = \min_{U,u} \tilde{\ell}(U, u) + C = -\frac{1}{2} \operatorname{tr}[\Lambda^2\Gamma^{-1}] + C$$

where C is a constant independent with θ. For any global minimum of $\tilde{\ell}$, we have $uU = \Gamma^{-1}$.

As above lemma construction, we denote the optimal solution as W^{*PV} and W^{*KQ}. Taking one solution as $U = \Gamma^{-1}, u = 1$, we observe the minimizer of global training loss is of the form:

$$W^{*PV} = \begin{pmatrix} 0_{d\times d} & 0_d \\ 0_d^\top & 1 \end{pmatrix}, W^{*KQ} = \begin{pmatrix} \Gamma^{-1} & 0_d \\ 0_d^\top & 0 \end{pmatrix}. \tag{2}$$

We then prove our main theory Theorem 1 in Section 3, we first re-state below:

Theorem 1. *Consider distinct tasks k and g with corresponding examples $\mathcal{S}_k, \mathcal{S}_g$, If two tasks have confined support, assume Assumption 1, with high probability, the model has the compositional ability. Moreover,*

$$Acc_\theta(\mathcal{S}_k) + Acc_\theta(\mathcal{S}_g) \le Acc_\theta(\mathcal{S}_{k\cup g}).$$

Proof of Theorem 1. WLOG, consider two simple tasks, $K = 2$. We have $x = (a, b)$, where $a \in \mathbb{R}^{d_1}, b \in \mathbb{R}^{d_2}, d_1 + d_2 = d$. Since x only has large values on certain dimensions, it's equivalent to just consider corresponding dimensions in w, i.e. for simple task 1, we have $w^{(1)} = (w_a, w_{\delta b})$, for simple task 2, we have $w^{(2)} = (w_{\delta a}, w_b)$.

We have $x \sim \Lambda$, where:

$$\Lambda = \begin{pmatrix} \Lambda_{\mathbb{K}\mathbb{K}} & \Lambda_{\mathbb{K}\mathbb{G}} \\ \Lambda_{\mathbb{G}\mathbb{K}} & \Lambda_{\mathbb{G}\mathbb{G}} \end{pmatrix}$$

- Task 1: $x = (a, 0_{d_2})^\top + (0, b_\delta)^\top$, $y = (w_a^\top a, 0) + (0, w_{\delta b}^\top b_\delta)$.
- Task 2: $x = (0_{d_1}, b)^\top + (a_\delta, 0_{d_2})^\top$, $y = (0, w_b^\top b) + (w_{\delta a}^\top a_\delta), 0)$.
- Composed task: $x = (a, b)^\top + (a_\delta, b_\delta)^\top$, $y = (w_a^\top a, w_b^\top b) + (w_{\delta a}^\top a_\delta, w_{\delta b}^\top b_\delta)$.

The form of E is,

$$E := \begin{pmatrix} a_1 & a_2 & \dots & a_N & a_q \\ b_1 & b_2 & \dots & b_N & b_q \\ y_1 & y_2 & \dots & y_N & 0 \end{pmatrix} + E_r \in \mathbb{R}^{(d+2)\times(N+1)}. \tag{3}$$

where E_r represents the values caused by residual dimensions whose entries bounded by δ.

Following Equation (4.3) in Zhang et al. (2023b), we have

$$EE^\top = \frac{1}{N}\begin{pmatrix} \sum_{i=1}^N a_i a_i^\top + a_q a_q^\top & \sum_{i=1}^N a_i b_i^\top + a_q b_q^\top & \sum_{i=1}^N a_i y_i^\top \\ \sum_{i=1}^N b_i a_i^\top + b_q a_q^\top & \sum_{i=1}^N b_i b_i^\top + b_q b_q^\top & \sum_{i=1}^N b_i y_i^\top \\ \sum_{i=1}^N y_i a_i^\top & \sum_{i=1}^N y_i b_i^\top & \sum_{i=1}^N y_i y_i^\top \end{pmatrix} + \delta \cdot o(EE^\top). \tag{4}$$

The W^{PV} can be presented in block matrix

$$W^{PV} = \begin{pmatrix} W_{11}^{PV} & W_{12}^{PV} & W_{13}^{PV} \\ (W_{21}^{PV})^\top & W_{22}^{PV} & W_{23}^{PV} \\ (W_{31}^{PV})^\top & (W_{32}^{PV})^\top & W_{33}^{PV} \end{pmatrix} \in \mathbb{R}^{(d_1+d_2+2)\times(d_1+d_2+2)} \tag{5}$$

We can apply Lemma F.1 into optimization and recall

$$W^{*KQ} = \begin{pmatrix} \Gamma_{all}^{-1} & 0_d \\ 0_d^\top & 0 \end{pmatrix}. \tag{6}$$

where $\Gamma_{all}^{-1} \in \mathbb{R}^{(d_1+d_2)\times(d_1+d_2)}$. Consider two tasks only related to disjoint dimension of x, we also have $\sigma(\Lambda_{\mathbb{K}\mathbb{G}}) = \sigma(\Lambda_{\mathbb{G}\mathbb{K}}) \leq \epsilon$. Denote

$$\Lambda = \tilde{\Lambda} + \Lambda_r$$

where

$$\tilde{\Lambda} = \begin{pmatrix} \Lambda_{\mathbb{K}\mathbb{K}} & \\ & \Lambda_{\mathbb{G}\mathbb{G}} \end{pmatrix}, \Lambda_r = \begin{pmatrix} & \Lambda_{\mathbb{K}\mathbb{G}} \\ \Lambda_{\mathbb{G}\mathbb{K}} & \end{pmatrix}$$

We apply Lemma F.1 Recall $\Gamma := \left(1 + \frac{1}{N}\right)\Lambda + \frac{1}{N}\operatorname{tr}(\Lambda) I_{d\times d} \in \mathbb{R}^{d\times d}$, we have:

$$\begin{aligned} \Gamma &= \left(1 + \frac{1}{N}\right)\tilde{\Lambda} + \frac{1}{N}\operatorname{tr}(\tilde{\Lambda}) I_{d\times d} + \left(1 + \frac{1}{N}\right)\Lambda_r \\ &= \tilde{\Gamma} + \Gamma_r \end{aligned}$$

where denote $\Gamma_r = \left(1 + \frac{1}{N}\right)\Lambda_r$. We have:

$$\Gamma^{-1} = \tilde{\Gamma}^{-1} - \tilde{\Gamma}^{-1}\Gamma_r\tilde{\Gamma}^{-1} + \mathcal{O}(\Gamma_r)$$

We denote

$$\tilde{\Gamma} = \begin{pmatrix} \Gamma_1 & 0 \\ 0 & \Gamma_2 \end{pmatrix},$$

where $\Gamma_1 = \left(1+\frac{1}{N}\right)\Lambda_{\mathbb{KK}} + \frac{1}{N}\operatorname{tr}(\Lambda)I_{d_1} \in \mathbb{R}^{d_1\times d_1}$ and $\Gamma_2 = \left(1+\frac{1}{N}\right)\Lambda_{\mathbb{GG}} + \frac{1}{N}\operatorname{tr}(\Lambda)I_{d_2} \in \mathbb{R}^{d_2\times d_2}$. Then we have;

$$\Gamma^{-1} = \begin{pmatrix} \Gamma_1^{-1} & 0 \\ 0 & \Gamma_2^{-1} \end{pmatrix} + A$$

where $\sigma(A) \leq 2m^2\epsilon$.

Then, It's similar to apply Lemma F.1 for pretraining separately into dimensions corresponding to different tasks. We solve similar to W_{KQ}.

we have:

$$f_\theta(E) = \begin{pmatrix} 0_{d_1\times d_1} & 0_{d_1\times d_2} & 0_{d_1\times 2} \\ 0_{d_2\times d_1} & 0_{d_2\times d_2} & 0_{d_2\times 2} \\ 0_{2\times d_1} & 0_{2\times d_2} & I_2 \end{pmatrix} EE^\top \begin{pmatrix} \Gamma_1^{-1} & 0_{d_1\times d_2} & 0_{d_1\times 2} \\ 0_{d_2\times d_1} & \Gamma_2^{-1} & 0_{d_2\times 2} \\ 0_{2\times d_1} & 0_{2\times d_2} & 0_{2\times 2} \end{pmatrix} \begin{pmatrix} a_q \\ b_q \\ 0 \end{pmatrix} + \tilde{A} \tag{7}$$

$$\hat{y}_q = \frac{1}{N}\left(\sum_{i=1}^N y_i a_i^\top, \sum_{i=1}^N y_i b_i^\top, \sum_{i=1}^N y_i y_i^\top\right) \begin{pmatrix} \Gamma_1^{-1}a_q \\ \Gamma_2^{-1}b_q \\ 0 \end{pmatrix} + v \tag{8}$$

$$= \left(\frac{1}{N}\sum_{i=1}^N y_i a_i^\top\right)\Gamma_1^{-1}a_q + \left(\frac{1}{N}\sum_{i=1}^N y_i b_i^\top\right)\Gamma_2^{-1}b_q + v \tag{9}$$

$$= \frac{1}{N}\begin{pmatrix} a_q^\top \Gamma_1^{-1}\sum_{i=1}^N y_i^{(1)} a_i \\ b_q^\top \Gamma_2^{-1}\sum_{i=1}^N y_i^{(2)} b_i \end{pmatrix} + v. \tag{10}$$

where $\tilde{A}$ representing residual matrix whose norm can be bounded by $\mathcal{O}(m^2\epsilon\delta)$ Recall $x \sim N(0,\Lambda)$, then with high probability each entry in v will be bounded by $Cm^2\delta\epsilon$ for some constant C.

WLOG, we write residual vectors as 0 vector for simplicity of notation, and only consider residuals for estimations $\hat{y}$. Note that composed example $x = (a,b)^\top$, $y = (w_a^\top a, w_b^\top b)$. For simplicity, we write $\hat{w}_a = \frac{1}{N}\Gamma_1^{-1}\sum_{i=1}^N y_i^{(1)} a_i$, similarly, $\hat{w}_b = \frac{1}{N}\Gamma_2^{-1}\sum_{i=1}^N y_i^{(2)} b_i$.

Given in-context examples from one simple task only, consider we have N examples from simple task 1, $\mathcal{S}_1 = \left[\{(a_i,0),y_i\}_{i=1}^N\right]$. We have $\hat{w}^{(1)} = (\hat{w}_a, 0_{d_2})$, $\hat{w}^{(2)} = (0_d)$, and we also have $\hat{y}_q = (\hat{y}_q^{(1)}, 0)^\top$, where $\hat{y}_q^{(1)} = a_q^\top\Gamma_1^{-1}\left(\frac{1}{N}\sum_{i=1}^N y_i^{(1)} a_i\right) + Cm^2\delta\epsilon$. We have $\mathrm{Acc}_\theta(\mathcal{S}_1) = \frac{\mathbb{1}\left(\widetilde{y}_q^{(1)} = y_q^{(1)}\right)}{2}$.

Similarly, for N in-context examples only from task 2, we have $\hat{w}^{(1)} = (0_d)$, $\hat{w}^{(2)} = (0_{d_1}, \hat{w}_b)$, $\hat{y}_q = (0, \hat{y}_q^{(2)})^\top$, where $\hat{y}_q^{(2)} = a_q^\top\Gamma_2^{-1}\left(\frac{1}{N}\sum_{i=1}^N y_i^{(2)} b_i\right) + Cm^2\delta\epsilon$. We have $\mathrm{Acc}_\theta(\mathcal{S}_2) = \frac{\mathbb{1}\left(\widetilde{y}_q^{(2)} = y_q^{(2)}\right)}{2}$.

Then we have $\mathcal{S}_{1\cup 2}$ contains $2N$ in-context examples from both tasks, specifically, we have N from task 1 and rest from task 2. We have $\hat{w}^{(1)} = (\hat{w}_a/2, 0_{d_2})$, $\hat{w}^{(2)} = (0_{d_1}, \hat{w}_b/2)$, $\hat{y}_q = (\hat{y}_q^{(1)}, \hat{y}_q^{(2)})^\top$.

Since $y_{\tau,q}^{(k)} = \mathrm{sgn}(\langle w_\tau, x_{\tau,q}\rangle)$, $\widetilde{y}_{\tau,q}^{(k)} = \mathrm{sgn}\left(\widehat{y}_{\tau,q}^{(k)}\right)$, following the proof of Lemma F.2, where the Acc_θ only concerns the direction of $\hat{w}$ and w, we have $\mathrm{Acc}_\theta(\mathcal{S}_{1\cup 2}) = \frac{\mathbb{1}\left(\widetilde{y}_q^{(1)} = y_q^{(1)}\right) + \mathbb{1}\left(\widetilde{y}_q^{(2)} = y_q^{(2)}\right)}{2}$.

Extending the above analysis into any of two simple tasks, when the composite task integrates them, we have

$$\mathrm{Acc}_\theta(\mathcal{S}_k) + \mathrm{Acc}_\theta(\mathcal{S}_g) \leq \mathrm{Acc}_\theta(\mathcal{S}_{k\cup g}). \tag{11}$$

□

We then prove Corollary 1 in Appendix E.1, we first restate it below.

Corollary 1. *If two tasks do not have confined support, there exists one setting which we have*

$$Acc_\theta(\mathcal{S}_k) = Acc_\theta(\mathcal{S}_g) = Acc_\theta(\mathcal{S}_{k\cup g}).$$

Proof of Corollary 1. WLOG, consider two simple tasks, $K = 2$. We have $x = (a, b)$, where $a \in \mathbb{R}^{d_1}, b \in \mathbb{R}^{d_2}, d_1 + d_2 = d$. Consider the setting where w also have the same active dimensions, i.e. for simple task 1, we have $w^{(1)} = (w_a, 0)$, for simple task 2, we have $w^{(2)} = (0, w_b)$.

We have $x \sim \Lambda$. Consider tasks are overlapping on all dimensions, where:

- Task 1: $x = (a^{(1)}, b^{(1)})^\top$, $y = (w_a^\top a^{(1)}, w_b^\top b^{(1)})$.
- Task 2: $x = (a^{(2)}, b^{(2)})^\top$, $y = (w_a^\top a^{(2)}, w_b^\top b^{(2)})$.
- Composed task: $x = (a, b)^\top$, $y = (w_a^\top a, w_b^\top b)$.

Similarly we have:

$$\hat{y}_q = \frac{1}{N}\left(\sum_{i=1}^N y_i a_i^\top, \sum_{i=1}^N y_i b_i^\top, \sum_{i=1}^N y_i y_i^\top\right)\begin{pmatrix}\Gamma_1^{-1} a_q \\ \Gamma_2^{-1} b_q \\ 0\end{pmatrix} \tag{12}$$

$$= \left(\frac{1}{N}\sum_{i=1}^N y_i a_i^\top\right)\Gamma_1^{-1} a_q + \left(\frac{1}{N}\sum_{i=1}^N y_i b_i^\top\right)\Gamma_2^{-1} b_q \tag{13}$$

$$= \frac{1}{N}\begin{pmatrix} a_q^\top \Gamma_1^{-1} \sum_{i=1}^N y_i^{(1)} a_i + b_q^\top \Gamma_2^{-1} \sum_{i=1}^N y_i^{(1)} b_i \\ a_q^\top \Gamma_1^{-1} \sum_{i=1}^N y_i^{(2)} a_i + b_q^\top \Gamma_2^{-1} \sum_{i=1}^N y_i^{(2)} b_i \end{pmatrix}. \tag{14}$$

Note that composed example $x = (a, b)^\top$, $y = (w_1^\top a, w_2^\top b)$.

When in-context examples from a simple task, we have N examples from simple task 1, $\mathcal{S}_1 = \left[\left\{(a_i^{(1)}, b_i^{(1)}), y_i\right\}_{i=1}^N\right]$, and $\hat{y}_q$ has the same form as Equation (14). Similarly for task 2.

Suppose $\mathcal{S}_{1\cup 2}$ contains $2N$ examples from both tasks, where N from task 1 and rest from task 2. We have

$$\hat{y}_q = \frac{1}{2N}\begin{pmatrix} a_q^\top \Gamma_1^{-1} \sum_{i=1}^N y_i^{(1)} a_i + b_q^\top \Gamma_2^{-1} \sum_{i=1}^N y_i^{(1)} b_i \\ a_q^\top \Gamma_1^{-1} \sum_{i=1}^N y_i^{(2)} a_i + b_q^\top \Gamma_2^{-1} \sum_{i=1}^N y_i^{(2)} b_i \end{pmatrix}. \tag{15}$$

We finish the proof by checking that Equation (14) and Equation (15) share the same direction. □

F.2 Proof of Compositional Ability with Model Scale

Here, we provide the proof of our conclusions in Theorem 2 in Appendix E.2 regarding model performance and model scale. We first introduce a lemma under the $K = 1$ setting.

F.2.1 Accuracy under $K = 1$

When $K = 1$, we can give an upper bound of the accuracy by Λ and Γ. Considering the optimal solution in Equation (2), we have a lemma of accuracy below.

Lemma F.2. *Consider $K = 1$ and $x_q \sim \mathcal{N}(0, I_d)$. When $N > C$, where C is a constant, we have*

$$\mathbb{E}_{w_\tau, x_1, \cdots, x_N}\left[Acc_\theta\right] \le \operatorname{tr}(\Gamma^{-1}\Lambda).$$

Proof of Lemma F.2. Since $K = 1$, the problem reduces to the linear regression problem in ICL. Consider the solution form in Lemma F.1, we have

$$\hat{y}_q = x_q^\top \frac{1}{N}\Gamma^{-1}\sum_{i=1}^N \langle w_\tau, x_i\rangle x_i$$

We re-write the form as $\hat{y}_q = x_q^\top \hat{w}$. Following Equation (4.3) in Zhang et al. (2023b), we have:

$$\hat{w} = \frac{1}{N}\Gamma^{-1}\sum_{i=1}^{N}\langle w_\tau, x_i\rangle x_i.$$

Recall the definition of Acc_θ and $y_{\tau,q}^{(k)} = \mathrm{sgn}(\langle w_\tau, x_{\tau,q}\rangle)$, $\widetilde{y}_{\tau,q}^{(k)} = \mathrm{sgn}\left(\widehat{y}_{\tau,q}^{(k)}\right) = \mathrm{sgn}(\langle \hat{w}, x_{\tau,q}\rangle)$, for any $\alpha > 0$, we have:

$$\mathbb{E}_{w_\tau, x_1,\cdots,x_N,x_q}\left[\mathrm{Acc}_\theta\right] = P\left(\langle x_q, w_\tau\rangle > 0, \langle x_q, \alpha\hat{w}\rangle > 0\right) + P\left(\langle x_q, w_\tau\rangle < 0, \langle x_i, \alpha\hat{w}\rangle < 0\right).$$

Denote hyperplane orthogonal to w as $\mathcal{P}_w$ and similar for $\mathcal{P}_{\hat{w}}$. Recall that x_q is independent of other samples. We have the expectation conditioned on $w_\tau, x_1, \cdots, x_N$ is the probability x_q falls out of the angle between $\mathcal{P}_w$ and $\mathcal{P}_{\hat{w}}$. Denote the angle between w and $\hat{w}$ as $\widetilde{\theta}$. As x_q is uniform along each direction (uniform distribution or isotropic Gaussian) then the probability is $1 - \frac{|\widetilde{\theta}|}{\pi}$ given $w_\tau, x_1, \cdots, x_N$. Then $\mathbb{E}_{w_\tau,x_1,\cdots,x_N}\left[\mathrm{Acc}_\theta\right] = \mathbb{E}_{w_\tau,x_1,\cdots,x_N}\left[1 - \frac{|\widetilde{\theta}|}{\pi}\right]$. Note that

$$\mathbb{E}_{w_\tau,x_1,\cdots,x_N}\left[\cos(\widetilde{\theta})\right] = \left\langle \frac{w_\tau}{\|w_\tau\|_2}, \frac{\hat{w}}{\|\hat{w}\|_2}\right\rangle.$$

As, we can choose α, w.l.o.g, we take $\|w_\tau\| = \|\hat{w}\| = 1$, then we have

$$\mathbb{E}_{w_\tau,x_1,\cdots,x_N}\left[\cos(\widetilde{\theta})\right] = \mathbb{E}_{w_\tau}\left[\mathbb{E}_{x_1,\cdots,x_N}\left[\langle w_\tau, \hat{w}\rangle | w_\tau\right]\right].$$

Given w_τ, we have

$$\begin{aligned}
E[\hat{w}|w_\tau] &= \frac{1}{N}\Gamma^{-1}\sum_{i=1}^{N} E\left[\langle w_\tau, x_i\rangle x_i | w_\tau\right] \\
&= \frac{1}{N}\Gamma^{-1}\sum_{i=1}^{N}\Lambda w_\tau \\
&= \Gamma^{-1}\Lambda w_\tau.
\end{aligned}$$

Then, we have

$$\begin{aligned}
E_{w_\tau}\left[\langle \hat{w}, w_\tau\rangle\right] &= \left\langle \Gamma^{-1}\Lambda w_\tau^\top, w_\tau\right\rangle \\
&= \mathrm{tr}(\Gamma^{-1}\Lambda).
\end{aligned}$$

Thus, we have

$$\mathbb{E}\cos(\widetilde{\theta}) = \mathrm{tr}(\Gamma^{-1}\Lambda) \tag{16}$$

$$\mathbb{E}\left[\mathrm{Acc}_\theta\right] = \mathbb{E}\left[1 - \frac{|\widetilde{\theta}|}{\pi}\right]. \tag{17}$$

Note the fact that when $\theta \leq \frac{\pi}{6}$, we have $1 - \frac{|\widetilde{\theta}|}{\pi} \leq \cos(\theta)$. Thus, as $N > C$ where C is constant, we have $\hat{w}$ and w_τ are closed and satisfy $\theta \leq \frac{\pi}{6}$. Then we get the statement. □

F.2.2 Model scale on composite tasks

Here we present proof for model scale and performance on composite tasks. Recall we consider the rank of W^{*PV} and W^{*KQ} as a measure of the model's scale.

We first introduce a lemma about U as an optimal full-rank solution.

Lemma F.3 (Corollary A.2 in Zhang et al. (2023b))**.** *The loss function $\tilde{\ell}$ in Lemma F.1 satisfies*

$$\min_{U\in\mathbb{R}^{d\times d}, u\in\mathbb{R}} \tilde{\ell}(U, u) = -\frac{1}{2}\,\mathrm{tr}[\Lambda^2\Gamma^{-1}], \tag{18}$$

where $U = c\Gamma^{-1}, u = \frac{1}{c}$ for any non-zero constant c are minimum solution. We also have

$$\tilde{\ell}(U, u) - \min_{U\in\mathbb{R}^{d\times d}, u\in\mathbb{R}} \tilde{\ell}(U, u) = \frac{1}{2}\left\|\Gamma^{\frac{1}{2}}\left(u\Lambda^{\frac{1}{2}}U\Lambda^{\frac{1}{2}} - \Lambda\Gamma^{-1}\right)\right\|_F^2. \tag{19}$$

As the scale of the model decreases, the rank of U also reduces, leading to an optimal reduced rank solution $\widetilde{U}$. Our findings reveal that this reduced rank $\widetilde{U}$ can be viewed as a truncated form of the full-rank solution U. This implies that smaller-scale models are essentially truncated versions of larger models, maintaining the core structure but with reduced complexity.

Recall Λ is the covariance matrix, we have eigendecomposition $\Lambda = QDQ^\top$, where Q is an orthonormal matrix containing eigenvectors of Λ and D is a sorted diagonal matrix with non-negative entries containing eigenvalues of Λ, denoting as $D = \operatorname{diag}([\lambda_1, \dots, \lambda_d])$, where $\lambda_1 \geq \cdots \geq \lambda_d \geq 0$. We introduce lemma below.

Lemma F.4 (Optimal rank-r solution)**.** *Recall the loss function $\tilde{\ell}$ in (Lemma F.1). Let*

$$U^*, u^* = \underset{U \in \mathbb{R}^{d\times d}, \operatorname{rank}(U) \leq r, u \in \mathbb{R}}{\arg\min} \tilde{\ell}(U, u). \tag{20}$$

Then $U^* = cQV^*Q^\top, u = \frac{1}{c}$, where c is any non-zero constant and $V^* = \operatorname{diag}([v_1^*, \dots, v_d^*])$ is satisfying for any $i \leq r, v_i^* = \frac{N}{(N+1)\lambda_i + \operatorname{tr}(D)}$ and for any $i > r, v_i^* = 0$.

Then, we proof the Lemma F.4

Proof of Lemma F.4. Note that,

$$\begin{aligned}
\underset{U \in \mathbb{R}^{d\times d}, \operatorname{rank}(U) \leq r, u \in \mathbb{R}}{\arg\min} \tilde{\ell}(U, u) &= \underset{U \in \mathbb{R}^{d\times d}, \operatorname{rank}(U) \leq r, u \in \mathbb{R}}{\arg\min} \tilde{\ell}(U, u) - \min_{U \in \mathbb{R}^{d\times d}, u \in \mathbb{R}} \tilde{\ell}(U, u) && (21)\\
&= \underset{U \in \mathbb{R}^{d\times d}, \operatorname{rank}(U) \leq r, u \in \mathbb{R}}{\arg\min} \left(\tilde{\ell}(U, u) - \min_{U \in \mathbb{R}^{d\times d}, u \in \mathbb{R}} \tilde{\ell}(U, u) \right). && (22)
\end{aligned}$$

Thus, we may consider Equation (19) in Lemma F.3 only. On the other hand, we have

$$\begin{aligned}
\Gamma &= \left(1 + \frac{1}{N}\right) \Lambda + \frac{1}{N} \operatorname{tr}(\Lambda) I_{d\times d} && (23)\\
&= \left(1 + \frac{1}{N}\right) QDQ^\top + \frac{1}{N} \operatorname{tr}(D) Q I_{d\times d} Q^\top && (24)\\
&= Q \left(\left(1 + \frac{1}{N}\right) D + \frac{1}{N} \operatorname{tr}(D) I_{d\times d} \right) Q^\top. && (25)
\end{aligned}$$

We denote $D' = \left(1 + \frac{1}{N}\right) D + \frac{1}{N} \operatorname{tr}(D) I_{d\times d}$. We can see $\Lambda^{\frac{1}{2}} = QD^{\frac{1}{2}}Q^\top$, $\Gamma^{\frac{1}{2}} = Q{D'}^{\frac{1}{2}}Q^\top$, and $\Gamma^{-1} = Q{D'}^{-1}Q^\top$. We denote $V = uQ^\top UQ$. Since Γ and Λ are commutable and the Frobenius norm (F-norm) of a matrix does not change after multiplying it by an orthonormal matrix, we have Equation (19) as

$$\begin{aligned}
\tilde{\ell}(U, u) - \min_{U \in \mathbb{R}^{d\times d}, u \in \mathbb{R}} \tilde{\ell}(U, u) &= \frac{1}{2} \left\| \Gamma^{\frac{1}{2}} \left(u \Lambda^{\frac{1}{2}} U \Lambda^{\frac{1}{2}} - \Lambda \Gamma^{-1} \right) \right\|_F^2 && (26)\\
&= \frac{1}{2} \left\| \Gamma^{\frac{1}{2}} \Lambda^{\frac{1}{2}} \left(uU - \Gamma^{-1} \right) \Lambda^{\frac{1}{2}} \right\|_F^2 && (27)\\
&= \frac{1}{2} \left\| {D'}^{\frac{1}{2}} D^{\frac{1}{2}} \left(V - {D'}^{-1} \right) D^{\frac{1}{2}} \right\|_F^2. && (28)
\end{aligned}$$

As W^{KQ} is a matrix whose rank is at most r, we have V is also at most rank r. Then, we denote $V^* = \arg\min_{V \in \mathbb{R}^{d\times d}, \operatorname{rank}(V) \leq r} \left\| {D'}^{\frac{1}{2}} D^{\frac{1}{2}} \left(V - {D'}^{-1} \right) D^{\frac{1}{2}} \right\|_F^2$. We can see that V^* is a diagonal matrix. Denote $D' = \operatorname{diag}([\lambda'_1, \dots, \lambda'_d])$ and $V^* = \operatorname{diag}([v_1^*, \dots, v_d^*])$. Then, we have

$$\begin{aligned}
&\left\| {D'}^{\frac{1}{2}} D^{\frac{1}{2}} \left(V - {D'}^{-1} \right) D^{\frac{1}{2}} \right\|_F^2 && (29)\\
=& \sum_{i=1}^{d} \left({\lambda'_i}^{\frac{1}{2}} \lambda_i \left(v_i^* - \frac{1}{\lambda'_i} \right) \right)^2 && (30)\\
=& \sum_{i=1}^{d} \left(\left(1 + \frac{1}{N}\right) \lambda_i + \frac{\operatorname{tr}(D)}{N} \right) \lambda_i^2 \left(v_i^* - \frac{1}{\left(1 + \frac{1}{N}\right) \lambda_i + \frac{\operatorname{tr}(D)}{N}} \right)^2. && (31)
\end{aligned}$$

As V^* is the minimum rank r solution, we have that $v_i^* \geq 0$ for any $i \in [d]$ and if $v_i^* > 0$, we have $v_i^* = \frac{1}{(1+\frac{1}{N})\lambda_i + \frac{\mathrm{tr}(D)}{N}}$. Denote $g(x) = \left(\left(1+\frac{1}{N}\right)x + \frac{\mathrm{tr}(D)}{N}\right) x^2 \left(\frac{1}{(1+\frac{1}{N})x + \frac{\mathrm{tr}(D)}{N}}\right)^2 = x^2 \left(\frac{1}{(1+\frac{1}{N})x + \frac{\mathrm{tr}(D)}{N}}\right)$. It is easy to see that $g(x)$ is an increasing function on $[0, \infty)$. Now, we use contradiction to show that V^* only has non-zero entries in the first r diagonal entries. Suppose $i > r$, such that $v_i^* > 0$, then we must have $j \leq r$ such that $v_j^* = 0$ as V^* is a rank r solution. We find that if we set $v_i^* = 0, v_j^* = \frac{1}{(1+\frac{1}{N})\lambda_j + \frac{\mathrm{tr}(D)}{N}}$ and all other values remain the same, Equation (31) will strictly decrease as $g(x)$ is an increasing function on $[0, \infty)$. Thus, here is a contradiction. We finish the proof by $V^* = uQ^\top U^* Q$. □

We then ready to prove the Theorem 2 in Appendix E.2, we first re-state it below.

Theorem 2. *Suppose a composite task satisfies confined support. Suppose we have $(x_1, y_1, \ldots, x_N, y_N, x_q)$ as an testing input prompt, and corresponding W where $y_i = Wx_i$. As rank r decreases, $\mathbb{E}_{W,x_1,\cdots,x_N}[Acc_\theta]$ will have a smaller upper bound.*

Proof of Theorem 2. We first prove in a simple task setting ($K = 1$), that the accuracy will have such a conclusion. By Lemma F.2, consider $x_q \sim \mathcal{N}(0, I_d)$. When $N > C$, where C is a constant, we have

$$\mathbb{E}_{w_\tau,x_1,\cdots,x_N}[\mathrm{Acc}_\theta] \leq \mathrm{tr}(\Gamma^{-1}\Lambda).$$

Recall Lemma F.4. WLOG, we take $c = 1$. We have

$$\begin{aligned}\mathrm{tr}(\Gamma^{-1}\Lambda) &= \mathrm{tr}\left(QV^*DQ\right) \\ &= \sum_{i=1}^{r} \frac{N}{N + 1 + \sum_{j=1}^{r} \frac{\lambda_j}{\lambda_i}},\end{aligned}$$

where second equation comes from Lemma F.4.

Under the confined support setting, the same conclusion holds since Equation (11) in the proof of Theorem 1.

□