Exploring Compositional Generalization of Large Language Models

Haoran Yang♠ **, Hongyuan Lu**♠ **, Wai Lam**♠ **Deng Cai**♡
♠The Chinese University of Hong Kong ♡Tencent AI Lab
{hryang, hylu, wlam}@se.cuhk.edu.hk jcykcai@tencent.com

Abstract

In this paper, we study the generalization ability of large language models (LLMs) with respect to compositional instructions, which are instructions that can be decomposed into several sub-instructions. We argue that the ability to generalize from simple instructions to more intricate compositional instructions represents a key aspect of the out-of-distribution generalization for LLMs. Since there are no specialized datasets for studying this phenomenon, we first construct a dataset with the help of ChatGPT, guided by the self-instruct technique. Then, we fine-tune and evaluate LLMs on these datasets. Interestingly, our experimental results indicate that training LLMs on higher-order compositional instructions enhances their performance on lower-order ones, but the reverse does not hold true. The code and data are available at https://github.com/LHRYANG/Compositional_Generalization.

1 Introduction

Large language models (LLMs) such as GPT3 (Brown et al., 2020), LLaMA (Touvron et al., 2023a) and LLaMA-2 (Touvron et al., 2023b) have demonstrated excellent multitask-solving abilities largely due to instruction tuning (Ouyang et al., 2022) which fine-tunes LLMs to follow diverse and natural instructions.

This study aims to advance the understanding of the instruction-tuning process, specifically focusing on the compositional generalization ability of LLMs. Compositional instructions can be vaguely defined as complex instructions that are divisible into several simpler sub-instructions. Several studies have delved into different aspects of compositionality with different interpretations of the above definition. For instance, Lake and Baroni (2018) found that on their proposed SCAN dataset, RNN performs poorly when testing on longer sequences or primitive commands unseen during training. Keysers et al. (2020) constructed a realistic dataset based on question-answering datasets and regarded the novel compounds i.e., new ways of composing the atoms of the train set, as the out-of-domain test set on which they found the RNN model fails to generalize compositionally. Finlayson et al. (2022) conducted experiments on their built regular expression matching classification dataset and found T5-based models (Raffel et al., 2020) struggle with non-starfree or bigger r-languages. Anil et al. (2022) examined length generalization in LLMs, revealing significant deficiencies in their generalization capabilities when fine-tuned on tasks with different lengths. Although length can be positively correlated with the degree of compositionality, the two are not equivalent. Zhou et al. (2023) used instruction decomposition as an inference-time method for performance enhancement. This approach only focuses on task-specific prompt design and does not involve fine-tuning, which provides a limited understanding of the compositional generalization of LLMs.

Different from the above works, which either build unrealistic datasets that do not necessarily translate to the real world, or construct domain-specific datasets, which are limited in the era of multitask-solving LLMs, we aim to analyze the compositionality of LLMs by fine-tuning them on **instructions** drawn from **general domains** and of different complexities. Due to the lack of existing datasets tailored for this purpose, we leverage ChatGPT[1] and the self-instruct technique (Wang et al., 2023) to construct suitable datasets. Specifically, we generate compositional instructions with different orders (an order-n instruction means that the instruction can be decomposed into n sub-instructions). Following this, we proceed to fine-tune and evaluate a popular LLM series, LLaMA (Touvron et al., 2023a), using these datasets.

[1] https://chat.openai.com

Our primary objective is to investigate the prospect of whether LLMs, once trained on instructions of a particular order, can generalize effectively to instructions of a different order. Our experiments present a fascinating outcome. When LLMs are trained on higher-order compositional instructions, they show an enhancement in performance when dealing with lower-order ones. However, the reverse situation, where LLMs are trained on lower-order instructions and then assessed on higher-order ones, does not yield the same improvement in performance. This discovery could pave the way for new directions in the fine-tuning strategies of LLMs, potentially leading to more efficient and effective models.

2 Data Collection

2.1 Concept of Compositional Instructions

There are different interpretations of compositionality. For example, in Lake and Baroni (2018), compositional generalization usually refers to the ability to combine primitives into structures in novel ways, as exemplified by the SCAN dataset. In HotPotQA (Yang et al., 2018), compositional questions require reasoning over multiple steps to arrive at the right answer. For instance, "Who was president in the year Justin Bieber was born" requires the model to first determine when Justin Bieber was born, and then who the president was that year. In this work, we define compositional instruction as one that can be decomposed into multiple sub-instructions or steps[2]. More specifically:

An instruction is compositional if it can be decomposed into $n (n > 1)$ sub-instructions. This instruction is also called a n-decomposition or order-n instruction.

This definition is well-suited for real-world complex instructions in general domains, particularly when combined with techniques for robots to follow natural language instructions step-by-step to complete a task.

Here are some examples of compositional instructions, "`Translate the following paragraph to English and summarise the translated paragraph`" is a 2-decomposition (order-2) instruction, and "`Extract all the names in the following paragraph and Count the frequency of each name appearing and order them based on alphabet`" is a 3-decomposition (order-3) instruction. If an instruction is not compositional (e.g., "`Write an article about Summer.`"), we call it a 1-decomposition (order-1) instruction.

2.2 Dataset Generation

We take the idea of self-instruct (Wang et al., 2023) to generate compositional instructions with some modifications. In this paper, we only consider n-decomposition instructions where n ranges from 1 to 4. The 1-decomposition instruction dataset Alpaca-52k (Taori et al., 2023) has already been generated. We verified that these are overwhelmingly 1-decomposition instructions by randomly inspecting 200 instructions. The details of the checking process can be found in Appx. A. As a result, our efforts are concentrated on generating 2/3/4-decomposition instructions.

Seed Instruction Generation Seed instructions play a vital role in ensuring the diversity and quality of the generated data. Generating hundreds of sensible compositional instructions, particularly of high orders, can be a challenging task for humans. To address this, we begin by soliciting some 2-decomposition instructions from the extensive Belle corpus (about 2M instructions) (Ji et al., 2023a,b). The soliciting step involves querying gpt-3.5-turbo (Prompt used and detailed steps are provided in Appx. B.), followed by human labeling. Using these 2-decomposition instructions as a base, we then prompt gpt-3.5-turbo (with temperature 0.7) to generate higher-order instructions, which are again subject to human labeling (similar to procedures in Appx. A.). The prompt input submitted to gpt-3.5-turbo is depicted in Figure 1. It's noteworthy that the gray section of the prompt is not utilized in generating seed instructions. We discover that this configuration can enhance the diversity of high-order seed instructions. This may be due to the fact that without the presence of order-$(i+1)$ examples, gpt-3.5-turbo is afforded a greater freedom of thought. Ultimately, we generate 159/89/112 seed instructions with order 2/3/4, respectively.

Full Dataset Generation We utilize the same prompt to generate the full dataset as illustrated in Figure 1, including the gray section. In particular, when generating order-$(i+1)$ instructions, we sample instructions of orders ranging from 2 to $i + 1$. These samples are drawn from both the seed set and the set already generated. Subsequently, we in-

[2] Due to the complexity of languages, it is difficult to provide a very precise definition. Please refer to the limitation section.

Prompt used to generate instrcutions

```
[concept of compositional instruction]
Below are examples of 2-decomposition
instructions:
[2-decomposition instrcutions]
...
Below are examples of i-decomposition
instructions:
[i-decomposition instructions]
Below are examples of (i + 1)-decomposition
instructions:
[i + 1-decomposition instrctuins]

Please generate some (i + 1)-decomposition
instructions:
```

Figure 1: Compositional instruction generation prompt. The text in color gray is not used during generating seed instructions to improve diversity.

corporate these samples into the prompt to further generate more order-$(i+1)$ instructions. Finally, we have a total of 7000 instructions for each order and we regard the output of gpt-3.5-turbo (temperature 0.7) for these instructions as the ground truth. 6000 of them are regarded as the training set, the remaining 1000 are regarded as the test set. Analysis and some examples of the dataset are provided in Appx. C.

3 Experiments

Our study focuses on investigating the generalization ability of LLMs. Specifically, for each order instruction training dataset, we fine-tune a model and subsequently evaluate the model on instructions of various orders to assess their performance and adaptability.

3.1 Setup

Models We conduct experiments on LLaMA. LLaMA (Touvron et al., 2023a) is a collection of autoregressive language models ranging from 7B to 65B. In this paper, we report the results of the 7B and 13B models. We take two different tuning methods, full-finetuning and parameter-efficient tuning. Specifically, for parameter-efficient tuning, we choose LoRA (Hu et al., 2022) which injects trainable rank decomposition matrices into each layer of the Transformer architecture while keeping the pre-trained model weights frozen.

Evaluation Metrics We report Rouge-L [3] and BLEU (averaged from BLEU-1 to BLEU-4) [4] to measure two different aspects of the generated text in comparison to the reference text generated by ChatGPT. The BLEU metric is employed to calculate precision, while the ROUGE score is used to quantify recall.

Implementation Details For full-tuning, we adopt the AdamW (Kingma and Ba, 2017) optimizer, and the learning rate is set to 2e-5. The epoch is set to 2 and we use the last checkpoint to conduct evaluation on the test set. For LLaMA-LoRA (parameter-efficient tuning), the learning rate is set to 3e-4 and the epoch is set to 3. We use the last checkpoint to evaluate.

3.2 Results

We specifically examine two types of generalizations: forward generalization and backward generalization. Forward generalization refers to training the models on lower-order compositional instructions and evaluating their performance on higher-order instructions. Conversely, backward generalization involves training on higher-order instructions and evaluating on lower-order instructions. We report the results of LLaMA and LLaMA-LORA in Table 1 and Table 2 for the 7B model and 13B model respectively.

Forward Generalization The diagonal entries denote the results obtained from evaluating order-i instructions using the model trained on the order-i training set. By comparing the upper triangle entries with the corresponding diagonal entries, we notice obvious performance degradation for both full-tuning and parameter-efficient tuning models. An illustrative example is the (order-1, order-3) scenario in Table 1, where the achieved performance is 18.5/8.35, while 22.0/11.89 is achieved when using the model trained on order-3 datasets. We also notice that as the models are trained on higher-order datasets, the discrepancy between the forward results and the diagonal results gradually diminishes. For instance, when evaluated on the order-4 dataset, the (order-1, order-4) performance is 20.9/10.12, exhibiting a larger performance gap than the performance of (order-3, order-4) 24.0/13.09, when compared with the desired result 24.2/13.16. The conclusion is that forward generalization is usually

[3] https://github.com/pltrdy/rouge
[4] https://github.com/mjpost/sacrebleu

	order-1	order-2	order-3	order-4
LLaMA	ROUGE-L/BLEU	ROUGE-L/BLEU	ROUGE-L/BLEU	ROUGE-L/BLEU
order-1	16.7/6.94	20.9/10.46	18.5/8.35	20.9/10.12
order-2	17.2/6.82	23.2/12.39	21.8/11.51	23.2/12.35
order-3	17.2/7.00	23.3/12.72	22.0/11.89	24.0/13.09
order-4	17.7/6.93	22.8/12.25	21.8/11.74	24.2/13.16
LLaMA-LoRA	ROUGE-L/BLEU	ROUGE-L/BLEU	ROUGE-L/BLEU	ROUGE-L/BLEU
order-1	13.6/5.28	19.5/9.28	20.3/9.61	19.4/9.20
order-2	13.6/5.53	20.9/10.83	22.9/11.97	22.5/12.06
order-3	13.3/5.05	21.0/10.57	23.5/12.66	22.6/12.45
order-4	13.6/5.36	20.9/10.62	23.9/12.78	22.9/12.45

Table 1: Results of forward generalization (upper triangle of diagonal) and backward generalization (lower triangle of diagonal). The i_{th} row and j_{th} column means the model is trained on order-i instructions and evaluated on order-j instructions.

	order-1	order-2	order-3	order-4
LLaMA-13b	ROUGE-L/BLEU	ROUGE-L/BLEU	ROUGE-L/BLEU	ROUGE-L/BLEU
order-1	21.0/8.01	21.4/10.65	21.2/9.73	21.4/10.86
order-2	21.4/8.19	24.7/13.28	23.0/11.16	22.5/11.78
order-3	22.1/8.74	23.5/13.32	23.6/11.94	23.3/12.71
order-4	21.2/7.84	25.1/12.31	23.4/12.28	23.7/12.07
LLaMA-LoRA-13b	ROUGE-L/BLEU	ROUGE-L/BLEU	ROUGE-L/BLEU	ROUGE-L/BLEU
order-1	15.5/6.21	20.1/9.54	19.2/8.57	19.9/9.05
order-2	17.1/7.26	22.5/11.8	21.86/10.9	22.9/11.73
order-3	16.7/6.62	22.2/11.76	22.3/11.28	23.1/12.00
order-4	16.2/6.57	21.9/11.63	22.3/11.36	23.0/11.77

Table 2: Results of forward generalization (upper triangle of diagonal) and backward generalization (lower triangle of diagonal) of LLaMA-13b.

negative but the gap can be gradually narrowed by training on higher-order datasets.

Backward Generalization By analyzing the lower triangle entries with the corresponding diagonal entries, we find that there is a considerable proportion of positive backward generalization (indicated by the numbers in brown color.). As an illustration, the LLaMA-7B model, trained on order-3 dataset yields improved performance (23.3/12.72) as compared to that trained on order-2 dataset when evaluated on order-2 test set (23.2/12.39). It should also be noted that this phenomenon is also observed in LLaMA-13B as shown in Table 2. In conclusion, our findings suggest that LLMs trained on higher-order datasets can often outperform their counterparts trained on lower-order datasets when evaluating on the same lower-order test set. This phenomenon, termed as positive backward generalization, underscores the potential benefits of using higher-order datasets for model training to achieve improved performance even on lower-order tasks.

Impact of Output Length The ground-truth output length in high-order training set is obviously longer than the length in low-order training set as shown in Figure 4 Appx. C. And the length of the generated output also has an impact on ROUGE and BLEU. A natural question that arises is how reliable are the aforementioned conclusions. From Table 1 and Table 2, we can observe that BLEU and ROUGE *often* exhibit the same trend rather than one metric increasing while the other decreases. This implies that improvements in these metrics are indicative of overall enhancement in the quality of the generated text, rather than a trade-off between different evaluation criteria. We also plot the average generation length for each order test set, as illustrated in Figure 2. We can see that generation length is largely related to the test set instead of the models trained on datasets with different orders. It reveals that the model trained on datasets with short/long ground-truth output can still generate outputs with reasonable length based on the complexity of the instructions.

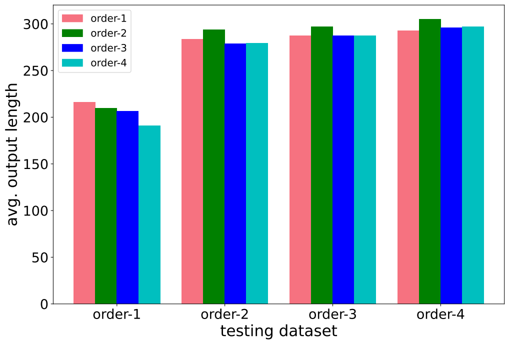

Figure 2: Average generation length comparison. The x-axis represents the test set with different orders while four colors represents four models tuned with different order training set.

	forward	backward
ROUGE (7B)	-7.6	3.1
ROUGE (13B)	-7.0	1.5
BLEU (7B)	-13.0	2.6
BLEU (13B)	-11.0	3.6

Table 3: Percentage (%) of performance drop and improvement. We only consider the positive forward and negative backward generalization.

3.3 Effect of Model Scale

To investigate whether the scale of the model can influence the degree of negative forward generalization and positive backward generalization, we compute the average percentage of performance deterioration and enhancement for the LLaMA-7b results (Table 1) and the LLaMA-13b results (Table 2). The statistics are presented in Table 3. Our analysis reveals that increasing the model scale indeed mitigates the extent of the performance drop in forward generalization (the 13B model has a small performance drop compared with the 7B model in both BLEU and ROUGE.). However, it remains inconclusive whether the model scale impacts the performance improvement in backward generalization. We leave it as a future work.

4 Conclusion

We studied the generalization ability of LLMs on compositional instructions. Our explorations highlighted the significant impact of the order of training instructions on performance. Specifically, while LLMs demonstrate negative forward generalization, they often exhibit positive backward generalization. Furthermore, we discern that a larger model scale can alleviate the negative forward generalization. We hope these discoveries will aid the research community in designing more effective instruction tuning strategies.

Limitations

In this work, we study the generalization ability of LLMs on compositional instructions. However, it is hard to precisely define the concept of compositional instruction due to the complexity of language and various interpretations. For example, “Write an article about Summer." can be further broken down into “Write an article" and "The article should be about Summer.". However, we regard it as a 1-decomposition instruction. Moreover, we use compositional instructions which are generated by ChatGPT. The diversity of these generated datasets may not be sufficiently high. The experiments are only conducted on LLaMA and the scale effect is also not thoroughly studied on much larger LLMs.

Ethics Statement

Due to the nature of language models, the generations may have offensive, toxic, unfair, or unethical biases. One can use post-process steps such as toxicity identification and fact checking to alleviate these issues.

References

Cem Anil, Yuhuai Wu, Anders Johan Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Venkatesh Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. 2022. Exploring length generalization in large language models. In *Advances in Neural Information Processing Systems*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Matthew Finlayson, Kyle Richardson, Ashish Sabharwal, and Peter Clark. 2022. What makes instruction learning hard? an investigation and a new challenge in a synthetic environment. In *Proceedings* *of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 414–426, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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

Yunjie Ji, Yong Deng, Yan Gong, Yiping Peng, Qiang Niu, Baochang Ma, and Xiangang Li. 2023a. Belle: Be everyone's large language model engine.

Yunjie Ji, Yong Deng, Yan Gong, Yiping Peng, Qiang Niu, Lei Zhang, Baochang Ma, and Xiangang Li. 2023b. Exploring the impact of instruction data scaling on large language models: An empirical study on real-world use cases. *arXiv preprint arXiv:2303.14742*.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. 2020. Measuring compositional generalization: A comprehensive method on realistic data. In *International Conference on Learning Representations*.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Brenden M. Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. Least-to-most prompting enables complex reasoning in large language models.

Prompt used to identify 2-decomposition instruction

```
[concept of compositonal instruction]
[1-decomposition instructions with
explanations]
[2-decomposition instructions with
explanations]

So please determine if [target instruction]
is a 2-decomposition instruction.]
```

Figure 3: Identify 2-decomposition instructions.

A Process of Labeling Alpaca-52k

We check whether most of the instructions in Alpaca-52k are 1-decomposition institutions. The annotation process is conducted by two annotators with high English proficiency. Firstly, we present them with the definition of compositional instruction as shown in Sec. 2.2. Then, for each order $n \in [1, 2, 3, 4]$, we give two example instructions. Finally, we ask the annotator to decide whether an instruction is compositional or non-compositional. If both annotators agree the instruction is compositional, then we label it as compositional instruction.

B Selecting 2-decomposition instructions from Belle Corpus

Due to the large size of the Belle Corpus (Ji et al., 2023a,b), it is impossible for human to label each instruction one by one. Therefore, we facilitate the powerful gpt-3.5-turbo to first distill some candidate 2-decomposition instructions. The prompt used to query gpt-3.5-turbo is shown in Figure 3. The temperature used for controlling generation is set to 0.7. We stop the running of gpt-3.5-turbo until 1000 candidate 2-decomposition instructions are collected. Then, we conduct the same step in Appx. A to manually label the valid 2-decomposition instructions. Finally, we obtain 159 2-decomposition instructions as the seed instructions.

C Dataset Analysis

Statistics We analyze (a) the proportion of instructions without input, (b) the average instruction length (excluding the input field), (c) the average input length, and (d) the average output length of different orders, as illustrated in Figure 4. We find (1) the percentage of instructions without input field are roughly close for instructions with different orders. (2) The average instruction length (the input field is not considered.) exhibits greater differences, where lower-order instructions are obviously shorter than high-order instructions. (3) The difference between the average input length for different orders (statistics on instructions that have an input filed) is not very significant. (4) The average output length of instructions with different orders is obviously different. This is expected, as higher-order instructions necessitate the completion of multiple tasks which should naturally result in longer instructions and outputs compared to lower-order prompts.

Diversity We also analyze the similarity of instructions for each order as shown in Figure 5. We plot the maximum Rouge-L score distribution all each order prompts test dataset. Specifically, for each prompt, we compute its Rouge-L score with each remaining prompt in the same datasets and take the largest value. A higher Rouge-L score indicates there is a prompt that is very similar to the current prompts. We find the similarity for order-2/3/4 instruction is slightly higher than order-1 instructions i.e., Alpaca-52k (Taori et al., 2023). We leave it as a future work to further improve the diversity.

Examples We provide some example instructions with different orders in Table 4.

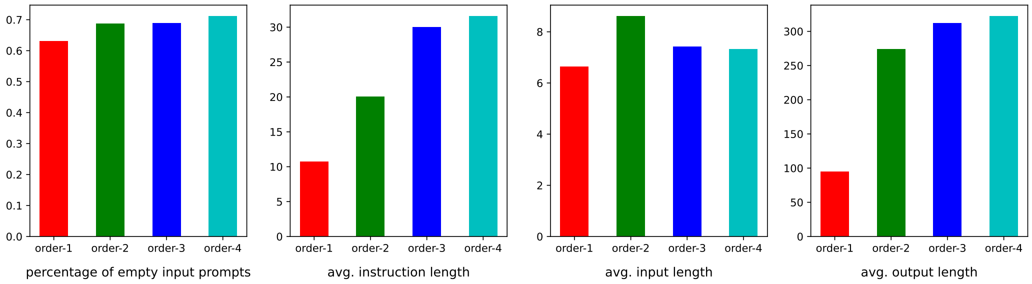

Figure 4: Basic statistics of the generated datasets, the order-1 dataset is built from alpaca while the other three are generated by ourselves. The main difference between these four datasets is in average instruction length and average. output length.

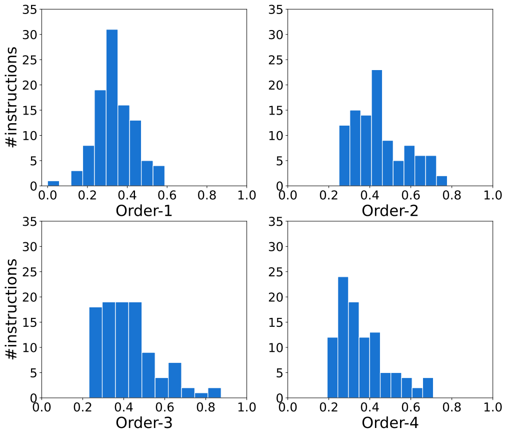

Figure 5: Rouge-L score distribution.

2-decomposition instructions	
instruction:	Write a research paper on the effects of social media on mental health, citing at least five academic sources.
input:	Topic: Effects of social media on mental health.
output:	Introduction \n Social media has become an integral part of modern society...
instruction:	Come up with at least 3 new menu items for a popular restaurant and provide a brief description of each dish, including ingredients and cooking method.
input:	None
output:	1. Spicy Mango Chicken: This dish ...
3-decomposition instructions	
instruction:	Extract all email addresses from the given text, sort them alphabetically, and count their occurrence.
input:	To contact us, please email support@company.com or sales@company.com. For general inquiries, you can also reach us at info@company.com.
output:	info@company.com (1)\n sales@company.com (1)\n support@company.com (1)
instruction:	Design a workout plan for a beginner. First, identify the fitness goals of the beginner. Second, suggest exercises that target all major muscle groups. Third, create a weekly schedule that incorporates cardio and strength training.
input:	None
output:	Fitness goals: The beginner wants to improve overall fitness, build strength, and lose weight...
4-decomposition instructions	
instruction:	Given a sample of employee resumes, identify the top three most qualified candidates, analyze their work experience and education, calculate the average salary range for the industry, and identify any potential red flags in the resumes.
input:	Resume1: Education=Bachelor’s degree, Work Experience=5 years, Salary Range=$50,000-$60,000; Resume2: Education=Master’s degree, Work Experience=3 years, Salary Range=$45,000-$55,000; Resume3: Education=Associate’s degree, Work Experience=8 years, Salary Range=$40,000-$50,000; Resume4: Education=High School diploma, Work Experience=10 years, Salary Range=$30,000-$40,000; Resume5: Education=Bachelor’s degree, Work Experience=2 years, Salary Range=$50,000-$60,000
output:	Based on the given information, the top three most qualified candidates are Resume1, Resume2, and Resume3 ...
instruction:	Develop a training program for new employees. Identify four key skills that new employees should learn, create training materials such as presentations and quizzes to test their knowledge, choose a training delivery method, and evaluate the effectiveness of your training program.
input:	None
output:	Training Program for New Employees \n Introduction:\n Congratulations on your new role as an employee of our company!...

Table 4: Some examples of the generated compositional instructions.