

Examining Compositional Behavior in Large Language Models

Jordan Tan
University of Utah

UUCS-24-004

School of Computing
University of Utah
Salt Lake City, UT 84112 USA

25 April 2024

Abstract

Large language models (LLMs) can be used to solve compositional tasks. These tasks involve knowledge of subtasks that must be combined to arrive at the correct answer. To better understand how LLMs approach compositional tasks, we conducted two compositional task experiments, multiplication and word list operation (WLO), on Flan-T5 and Llama 2-Chat models. We found that adding in-context examples of related skills did not improve any model's ability to perform the main task, regardless of whether the model could perform the related skills well.

EXAMINING COMPOSITIONAL BEHAVIOR
IN LARGE LANGUAGE MODELS

by
Jordan Tan


A Senior Honors Thesis Submitted to the Faculty of
The University of Utah
In Partial Fulfillment of the Requirements for the
Honors Degree in Bachelor of Science

In
Computer Science

Approved:



Vivek Srikumar
Thesis Faculty Supervisor



Mary Hall
Director, Kahlert School of Computing



Thomas C. Henderson
Honors Faculty Advisor

Monisha Pasupathi, PhD
Dean, Honors College

May 2024

Copyright © 2024

All Rights Reserved

ABSTRACT

Large language models (LLMs) can be used to solve compositional tasks. These tasks involve knowledge of subtasks that must be combined to arrive at the correct answer. To better understand how LLMs approach compositional tasks, we conducted two compositional task experiments, multiplication and word list operation (WLO), on Flan-T5 and Llama 2-Chat models. We found that adding in-context examples of related skills did not improve any model’s ability to perform the main task, regardless of whether the model could perform the related skills well.

CONTENTS

ABSTRACT.....	ii
INTRODUCTION	1
BACKGROUND: LARGE LANGUAGE MODELS	3
CAN COMPOSITIONALITY BE LEARNED IN-CONTEXT?.....	7
EXPERIMENT SETUP	9
RESULTS	15
MULTIPLICATION.....	16
WORD LIST.....	24
DISCUSSION & CONCLUSION	28
REFERENCES	30

INTRODUCTION

Recently, transformer-based large language models (LLMs) have become widely popular. One of the popular commercial LLMs used today is OpenAI’s ChatGPT¹, and LLMs that are open-source and are used in academic research include Flan-T5 and Llama 2. They have prolific usage in natural language processing (NLP) tasks, search engines², and code generation³. With their ability to perform a wide variety of tasks, LLMs are sometimes presented in academic literature and popular media as having “sparks of artificial general intelligence” [1].

Despite their ability to compose intricate responses, large language models have limited capability for compositional reasoning. These tasks are complex enough that they require multiple subtasks to complete them [2]. However, little is known about the causes of these limitations in LLMs. In this work, we explore whether compositionality can be learned in-context.

In-context learning is a popular technique to instruct large language models, where task examples are added to a model’s input. We wanted to determine if prompting an LLM with in-context examples of subtasks can improve compositionality. Compositional tasks require knowledge of multiple skills to derive the answer for the main task. Any intelligent system appropriately trained in the underlying skills can improve at a compositional task, but can today’s academic-scaled LLMs do so?

Let us consider an example of a compositional task: multiplication. It is a compositional task because it requires the following subtasks: single-digit multiplication,

¹ <https://openai.com/chatgpt>

² <https://www.bing.com/chat>

³ <https://github.com/features/copilot>

carry, addition, and concatenation. For an intelligent system to solve any multiplication problem, it must know how to perform these skills and compose them together to form the correct answer. If the system struggled to perform multiplication due to poor performance at a skill, then teaching it the skill would help it perform multiplication.

We conducted experiments on two tasks: multiplication and a more difficult compositional task involving string operations on lists of words. We focused our experiments on several models from two LLM families: Flan-T5 [3] and Llama 2-Chat [4]. Our results showed that the skills can be learned for both tasks, with larger models performing better. However, both models struggled at the compositional task, with Flan-T5 performing worse than Llama 2-Chat overall.

BACKGROUND: LARGE LANGUAGE MODELS

Large language models are machine learning models that can be used for text generation. Given a sequence of words, a language model tokenizes the sequence and returns probabilities for the next token after the sequence. To achieve good predictions, they are trained on an abundance of raw textual data based on a training objective. An example of a training objective is standard language modeling, where the model is tasked to optimize the probability of predicting text from the training data [5]. Perplexity is a metric for the quality of language models, and a lower perplexity corresponds with the model being more confident in its prediction based on the previous tokens [6].

There are many ways of obtaining the next n tokens in text generation. Greedy decoding considers the most probable token at each output, but this can lead to “repetitive and short output sentences” [7]. While considering all tokens at each step would produce the most probable sequence, this search is impractical due to its exponentially growing size. Beam search constrains the decoding process by considering the top b hypotheses at each output step [7]. This addresses the drawbacks of greedy decoding and maintains some benefits of exploring different paths. One modification to these strategies is to sample from the distribution rather than considering the highest probable tokens, increasing the diversity of outputs.

However, training a model for language modeling is not enough to use it for an NLP task. To obtain satisfactory results, one must provide context to the model about the downstream task. One way is to fine-tune a model to respond to instructions for the task [5]. While fine-tuning can improve a model’s performance on a downstream task, it requires a labeled dataset of instruction and output pairs, which can be difficult to procure

for certain tasks. To address this, models can be instruction-tuned on a large dataset of instruction-output pairs to follow instructions [8]. Instead of fine-tuning a model, one can engineer prompts for a model to obtain an answer to a downstream task. For instance, providing in-context examples of the task at hand can prime an LLM to respond to a query better [9]. With prompting, the terms few-shot, one-shot, and zero-shot correspond to the number of in-context examples given in a prompt: few-shot means having k in-context examples, one-shot means having one in-context example, and zero-shot means having no in-context examples [10].

Sentiment analysis is an example of an NLP task that one can solve using a large language model. It is a text classification problem where the goal is to predict the sentiment (attitude) of a given text. A popular dataset for this task is the Stanford Treebank Dataset (SST) [11], where extracted text from movie reviews is labeled as positive or negative. To solve this problem, one can use an LLM and construct a prompt. As an example, the prompt can look like this:

Classify the text sentiment as positive or negative.

Text: a technically superb film

Sentiment:

If the model was instruction-tuned, then fine-tuning a model on prompts given in this format should not be necessary; the model should know sentiment analysis because it is a common task in NLP. With the information provided, the model should produce “positive” over “negative” if it is more probable, but it may be necessary to provide additional context to the model. This can be done in the form of an in-context example. A one-shot prompt could then look like this:

Classify the text sentiment as positive or negative.

Text: contains no wit , only labored gags

Sentiment: negative

Classify the text sentiment as positive or negative.

Text: a technically superb film

Sentiment:

Note that there are many ways to format a prompt, and one may need to perform prompt engineering to find one that works well for a particular model.

Large language models of today use a transformer-based architecture. A transformer is a machine learning model that uses multi-head attention layers and positional encoding to process textual data [12]. The multi-head attention layers use attention, which is a mechanism that allows the model to focus on specific words in an input. Positional encoding is necessary for the model to understand which words come before or after other words. There are three types of transformers: encoder models, decoder models, and encoder-decoder models. Encoder-decoder models, such as Flan-T5, first generate an encoding of the input and uses a separate decoder to generate an output from the encoding [5]. Decoder models, such as Llama 2, feed the input into the decoder and then generate text that continues from the input [5].

For the experiments, we used two large language model families built upon the transformer architecture: Flan-T5 and Llama 2-Chat. Text-to-Text Transfer Transformer (T5) is an encoder-decoder model that underwent multi-task learning. It was trained on the Colossal Clean Crawled Corpus (C4), a large English corpus extracted from the web,

and on “a mixture of tasks” [13]. Flan-T5 builds upon T5 by incorporating instruction-tuning into its development, where it learned 1.8K tasks and how to do chain-of-thought [3].

Large Language Model Meta AI 2 (LLaMA 2) is a decoder model that updates Llama 1 by “train[ing] on a new mix of publicly available data,” doubling the context length, and more [4]. Llama 2-Chat is “a fine-tuned version of Llama 2 that is optimized for dialogue use cases” [4, p. 2]. The optimization was done using reinforcement learning with human feedback (RLHF), which is a process that involves maximizing a reward model trained to give higher scores to text preferred by humans [4]. Flan-T5 and Llama 2-Chat are distinct despite both models using a transformer-based architecture, and they provide a good mix for these experiments.

CAN COMPOSITIONALITY BE LEARNED IN-CONTEXT?

Compositionality is a topic that appears in different contexts, including semantics and mathematics. Generally, compositionality is the notion that “[t]he meaning of a complex expression is determined by its structure and the meanings of its constituents” [14]. While there is debate over the existence of compositionality in natural languages, compositionality can be purposely built into artificial languages, such as programming languages [15]. In this paper, we consider compositionality in a task-based context. One can view compositionality in terms of tasks for an overall skill.

Let us consider cooking as an example. If one is following a recipe, each step is a task that involves a skill. To cook well, one needs to know a variety of skills, such as selecting and measuring ingredients, cutting vegetables or meat, and frying, boiling, or baking. Even with a fixed set of ingredients to select and cooking methods, there are an endless amount new dishes one can make. These skills are separate from one another, and they are transferable. If one knows how to create a cake, then creating a cupcake would not be too difficult.

Compositional tasks often require a step-by-step approach to solving the task, which large language models can mimic. One idea to improve a model’s ability to reason is through chain-of-thought, which primes an LLM to respond to a query by mimicking how a human would approach solving a problem [16]. For instance, consider the following question: “If John purchased 2 watermelons for \$2 and 3 oranges for \$1, how much did he pay?” Instead of responding directly with the answer, the LLM could respond with this: “Let’s think this step-by-step. John purchased 2 watermelons for \$2, which is $2 * \$2 = \4 . John purchased 3 oranges for \$1, which is $3 * \$1 = \3 . Therefore,

the total is $\$4 + \$3 = \$7$.” Wang et al. improve on chain-of-thought reasoning with self-consistency, which can produce more accurate responses by sampling “over diverse reasoning paths,” rather than consider one path like chain-of-thought [17]. Because chain-of-thought and self-consistency can improve task performance, LLMs seem to exhibit or mimic some level of reasoning.

In-context learning can improve a model’s performance on a task, so it is productive to see if this applies to compositional tasks. Brown et al. trained GPT-3, “an autoregressive language model with 175 billion parameters” [10]. They experimented with prompting the model and found that few-shot prompting could outperform the state-of-the-art models on various datasets. Their findings reinforce the idea that in-context learning and prompt engineering are viable strategies for performing downstream tasks. If a task is compositional and models exhibit compositionality, in-context learning could help models by giving additional information about the skills. If this is the case, we would expect improvement in a model’s performance on the task. Furthermore, we would expect that in-context examples of skills that are not relevant to a task should offer no benefit to a task, either worsening or maintaining a model’s performance.

EXPERIMENT SETUP

For a compositional task, we define a main task and its subtasks. From Flan-T5, the Small, Base, Large, XL, and XXL models were used. From Llama 2, the Chat 7B and Chat 13B models were used. Each model was tested on the main task and subtask based on prompts. The prompt formats used for Flan-T5 are shown below in Figures 1-6. Llama 2-Chat used slightly different prompts because it uses additional instruction tags.

However, the instructions and questions remained the same. The prompts consisted of a worded instruction and a question, and each experiment was given an in-context example. Additionally, the main task was given an in-context example of the subtask. As a control, the main task was given an in-context example of a task unrelated to the main task. These tasks were subtraction, exponentiate, and string reversal. Each experiment was repeated with five different in-context examples, and the experiment was repeated five times with the same in-context example.

Q: Carry the digit from the tens place.
 27 // 10 =
 A: 2

Q: Carry the digit from the tens place.
 44 // 10 =
 A:

1a: Example of the carry prompt for Flan-T5.

Q: Concatenate the numbers.
 1 & 7 & 5 =
 A: 175

Q: Concatenate the numbers.
 2 & 3 & 3 =
 A:

1b: Example of the concatenation prompt for Flan-T5.

Q: Multiply two numbers.
 $4 * 9 =$
A: 36

Q: Multiply two numbers.
 $9 * 0 =$
A:

1c: Example of the multiplication of single digits prompt for Flan-T5.

Q: Add two numbers.
 $22 + 1 =$
A: 23

Q: Add two numbers.
 $18 + 41 =$
A:

1d: Example of the summation prompt for Flan-T5.

Figure 1: Example of skill prompts for the multiplication task for Flan-T5

Q: Multiply two numbers.
 $38 * 98 =$
A: 3724

Q: Multiply two numbers.
 $15480 * 75948 =$
A:

Figure 2: Example of the multiplication prompt for Flan-T5.

Q: Add two numbers.
 $38 + 98 =$
A: 136

Q: Multiply two numbers.
 $38 * 98 =$
A: 3724

Q: Multiply two numbers.
 $15480 * 75948 =$
A:

Figure 3: Example of the primed multiplication prompt for Flan-T5.

The multiplication task evaluated a model’s ability to perform reasoning with numerical tasks. For the multiplication experiment, the main task was multiplication of up to 5 digits. For up to 2-digit multiplication, all combinations were considered. For up to 3 to 5-digit multiplication, 1K problems were sampled for each digit. This resulted in 13K problems in total. The main task was composed of the following subtasks: multiplication of single digits (100 problems), carrying of the tens digit (90 problems), addition of up to two digits (10K problems), and concatenation of up to three digits (1.1K problems). All numbers considered were positive numbers.

Q: Concatenate the two word lists.
Concatenate of tyr, rectus, ki and chilli, vogul, bps is what?
A: tyr, rectus, ki, chilli, vogul, bps

Q: Concatenate the two word lists.
Concatenate of gaba, craved, ritz and krone, duenna, zoic is what?
A:

4a: Example of the concatenation prompt for Flan-T5.

Q: Uppercase all of the words in the list.
Uppercase of tyr, rectus, ki is what?
A: TYR, RECTUS, KI

Q: Uppercase all of the words in the list.
Uppercase of gaba, craved, ritz is what?
A:

4b: Example of the uppercase prompt for Flan-T5.

Q: Remove the first word in the list.
Remove first of tyr, rectus, ki is what?
A: rectus, ki

Q: Remove the first word in the list.
Remove first of gaba, craved, ritz is what?
A:

4c: Example of the remove first prompt for Flan-T5.

Figure 4: Example of skill prompts for the WLO task for Flan-T5.

Q: Perform these operations in order: uppercase all of the words in the first word list, remove the first word in the second word list, and concatenate the two word lists.
Using tyr, rectus, ki and chilli, vogul, bps gives what?
A: TYR, RECTUS, KI, vogul, bps

Q: Perform these operations in order: uppercase all of the words in the first word list, remove the first word in the second word list, and concatenate the two word lists.
Using gaba, craved, ritz and krone, duenna, zoic gives what?
A:

Figure 5: Example of WLO prompt for Flan-T5.

Q: Uppercase all of the words in the list.
Uppercase of tyr, rectus, ki is what?
A: TYR, RECTUS, KI

Q: Perform these operations in order: uppercase all of the words in the first word list, remove the first word in the second word list, and concatenate the two word lists.
Using tyr, rectus, ki and chilli, vogul, bps gives what?
A: TYR, RECTUS, KI, vogul, bps

Q: Perform these operations in order: uppercase all of the words in the first word list, remove the first word in the second word list, and concatenate the two word lists.
Using gaba, craved, ritz and krone, duenna, zoic gives what?
A:

Figure 6: Example of primed WLO prompt for Flan-T5.

The word list task evaluated a model’s ability to perform string operations on lists of words. The main task was a word list operation over two lists of words that consisted of the following tasks in order: uppercasing all of the words in the first list, removing the second word from the second list, and concatenating the two lists together. Each task had 1K problems containing a sample of words from a filtered WordNet corpus, where the words considered did not contain digits, did not contain underscores, were six characters or fewer, and were not obscenities.

Each prompt was given to the models to generate a response. Multinomial beam search with five beams was used. The correct answer was searched in each generated response in different ways. For the multiplication experiments, the response was stripped of commas, and the last number in the generated response was considered as the answer. For the WLO experiments, the answer was searched in the generated response such that

the final word list was in the response in order, and an additional check was used to ensure that the answer was not contained in a sublist of a larger list.

RESULTS

We aim to answer the following research questions in this section:

1. How well do the models perform at the given tasks and skills?
2. Does model parameter size affect the model's performance at the task?
3. If a model is given an in-context example of a skill relevant to a task, does this improve the model's performance at that task?

After conducting the experiments, we scored each experiment based on a scoring function. The results are shown below as box-and-whisker plots, representing each experiment's five average accuracies using different in-context examples. For each experiment, the accuracy of skills and the accuracy of the main task were measured. There are three variants for the experiments on the main task: original, primed, and control. The original experiment is performed with the main task prompt. The primed experiments contained an additional in-context example of a relevant skill to the main task prompt, and the control experiments contained an irrelevant in-context example instead.

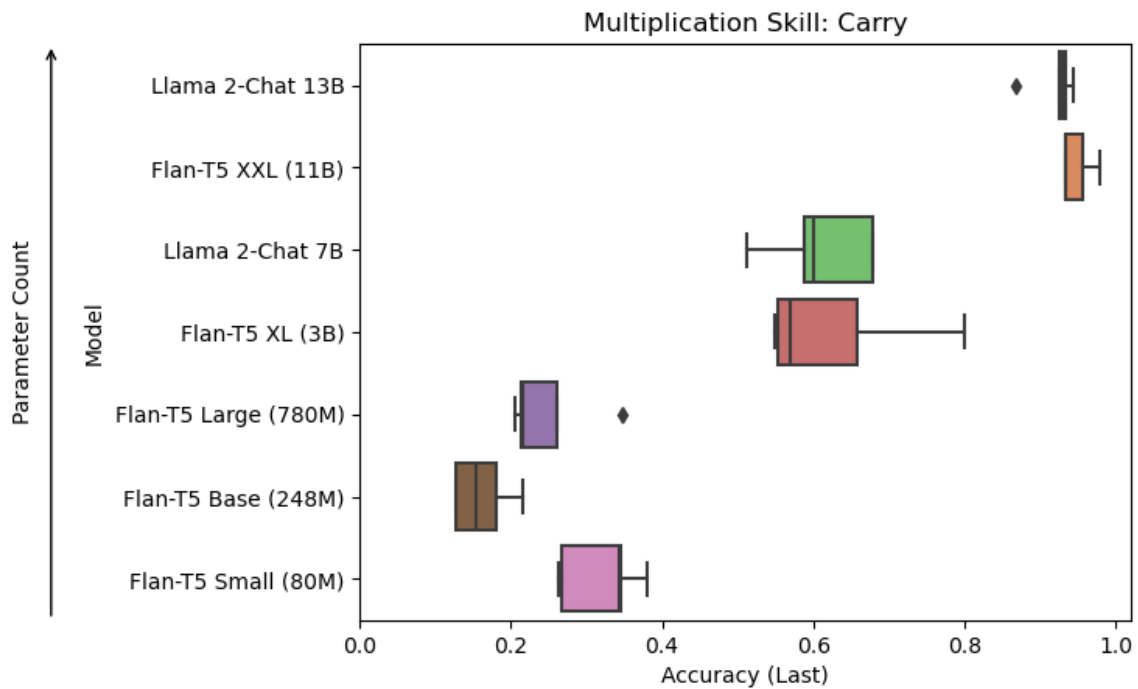
Flan-T5 XXL, Llama 2-Chat 7B, and Llama 2-Chat 13B can perform the skills needed for multiplication. However, all Flan-T5 models had near zero accuracy for the WLO task, so only the Llama 2-Chat models are shown in Figure 10. This indicates that Llama 2-Chat had better training in responding to these questions than Flan-T5. A general trend from the results was that larger models tended to perform better than smaller models (ignoring Flan-T5 for the WLO experiment). This coincides with the general trend of larger models performing better than smaller models [10]. However, there are some differences between the model families

chosen. Llama 2-Chat models performed better than their Flan-T5 counterparts in the main tasks and most skills (XL and XXL, respectively).

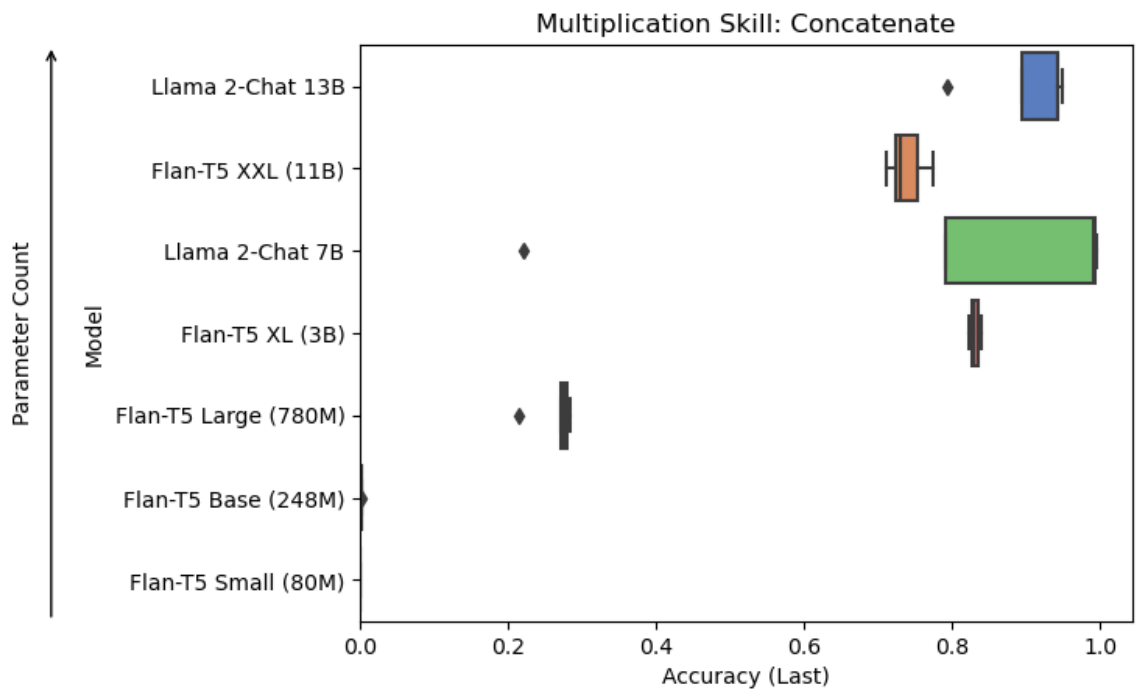
Additional information in a prompt can do one of three things: (1) it can improve the model's response by giving it useful information, (2) it can have no effect on the model's response and is ignored, or (3) it can worsen a model's response by adding noise and causing confusion. If large language models can learn a skill through in-context examples, then one would expect an improvement in accuracy by introducing in-context examples of relevant skills. However, the results disprove this notion, as experiments with primed prompts did not consistently improve. Furthermore, we expected that control experiments would either have no effect or worsen model performance, but there were cases where it improved accuracy. This suggests that no clear effect exists on how in-context examples for different skills can change model performance on a particular task.

MULTIPLICATION

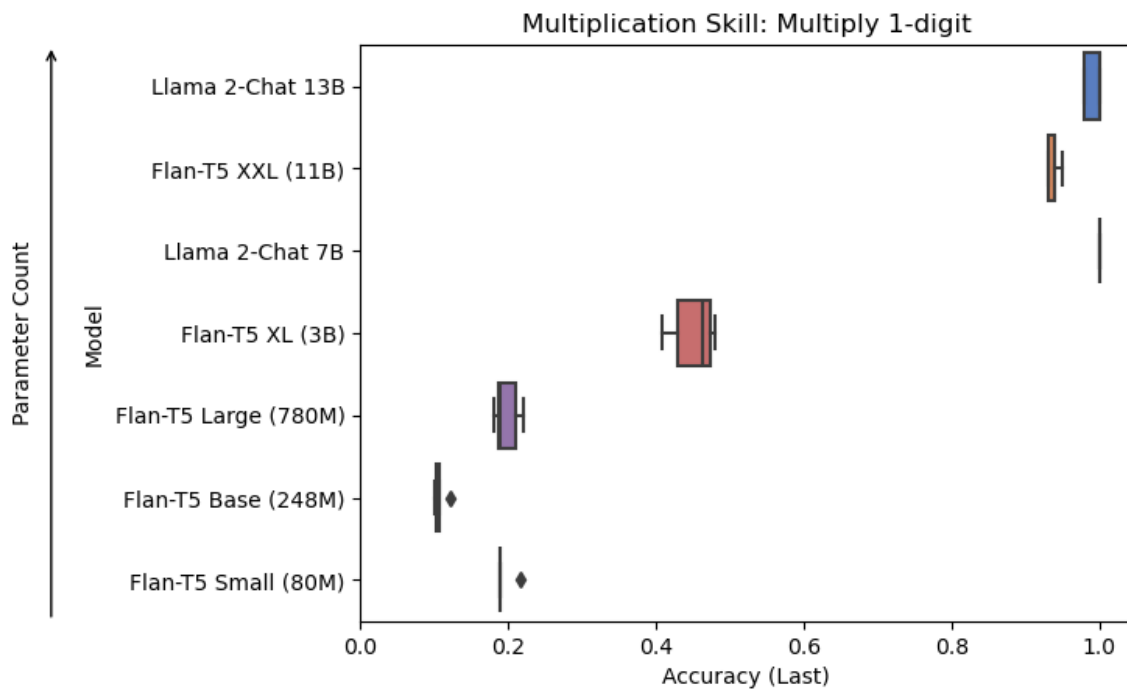
For the multiplication task, the models were evaluated on the following skills: carrying, concatenation, multiplication of single digits, and summation. The models were evaluated on multiplication, with and without being primed on a skill. Figure 7 shows the results of the skills across all models used in the experiment. Figure 8 shows the results of the main task across different prompts.



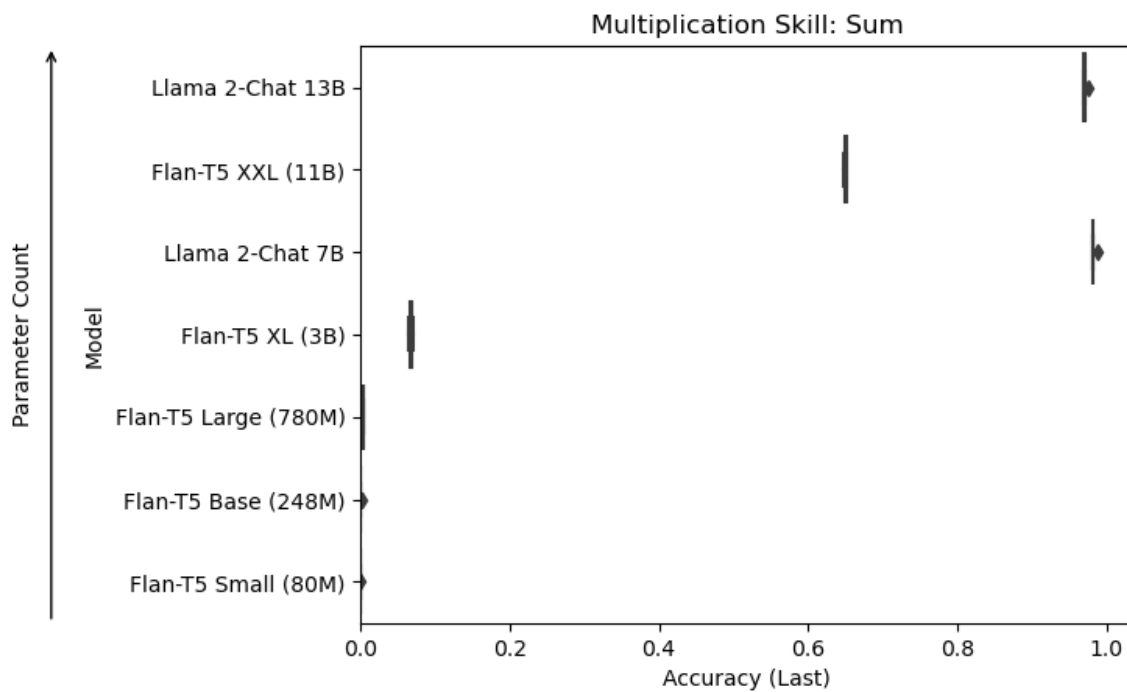
7a: Models of increasing size evaluated on accuracy for the carry skill.



7b: Models of increasing size evaluated on accuracy for the concatenation skill.



7c: Models of increasing size evaluated on accuracy for the multiplication of single digits skill.

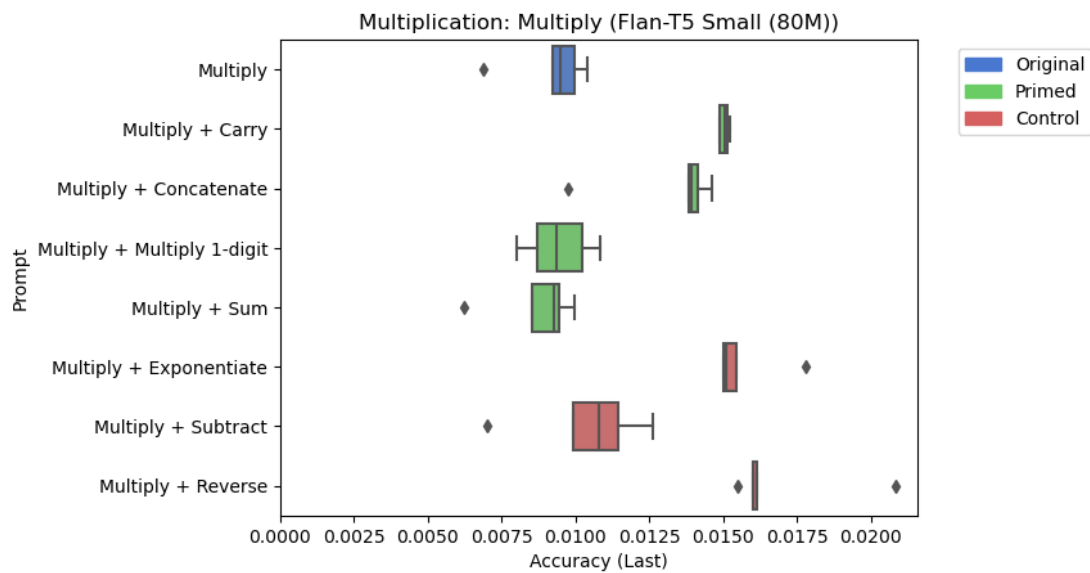


7d: Models of increasing size evaluated on accuracy for the summation skill.

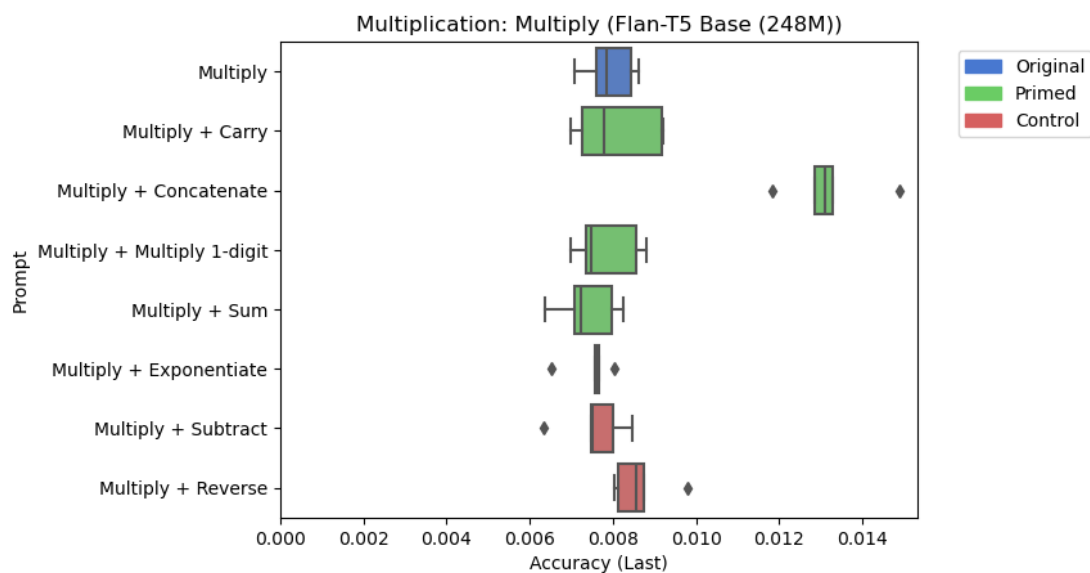
Figure 7: Results for skills used in multiplication.

Results were consistent across each skill, with the larger models performing better than the smaller models. Flan-T5 XXL, Llama 2-Chat 7B, and Llama 2-Chat 13B were the models that performed the skills well. With the Llama 2-Chat models, they occasionally responded with more than the eventual answer, producing a chain-of-thought response even without being prompted to. For the concatenation task, Llama 2-Chat 7B sometimes mistakenly appended zeros to the start of its answer. Llama 2-Chat 13B sometimes mistakenly skipped the first number for its answer if the first number was zero.

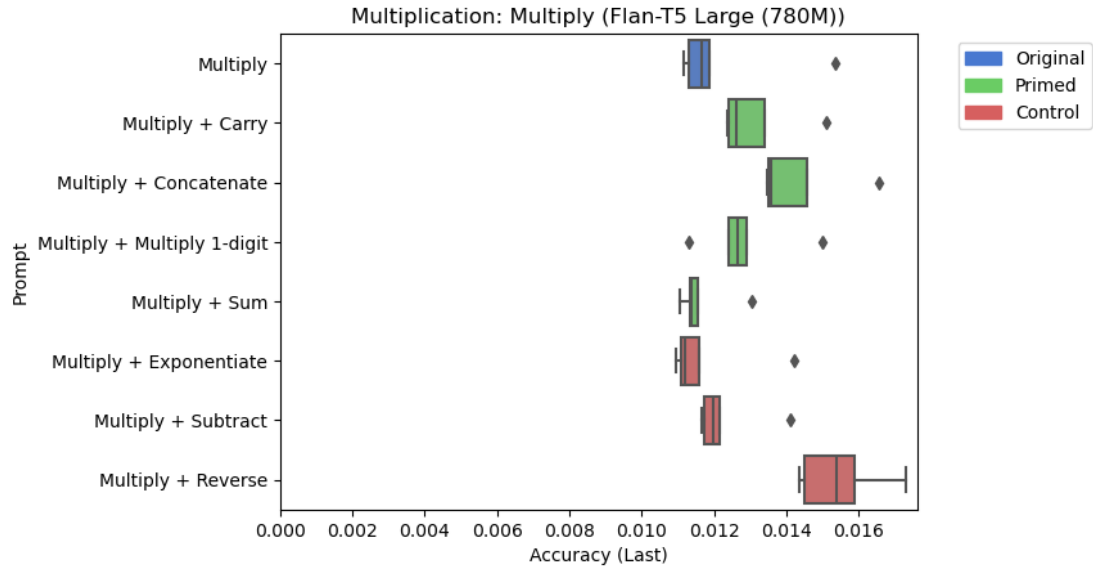
Smaller models tended to perform poorly due to various issues in their responses. For the carry skill, Flan-T5 Small, Flan-T5 Base, and Flan-T5 Large tended to respond with the answer for the in-context example rather than the question at hand. For the concatenation skill, Flan-T5 Base frequently resorted to repeating the question. For the sum skill, Flan-T5 Small and Flan-T5 Base frequently produced negative numbers in their answers, which would not be possible in this context because only positive numbers were used in the experiments. For the multiplication of single digits skill, Flan-T5 Small, Flan-T5 Base, and Flan-T5 Large frequently produced single-digit answers, even for questions that would result in a two-digit answer.



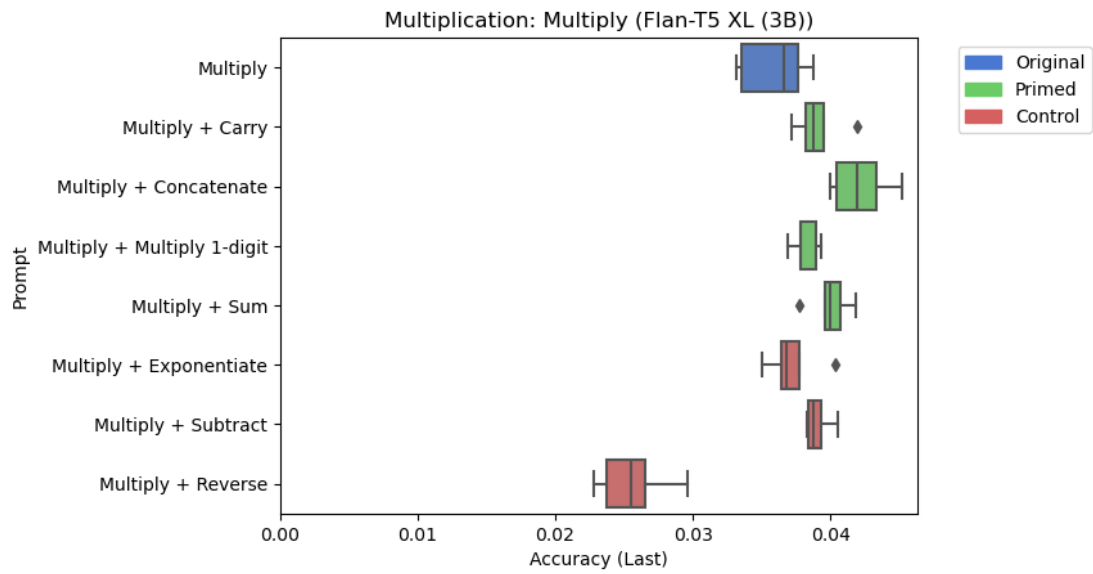
8a: Flan-T5 Small's accuracy for multiplication using different prompts.



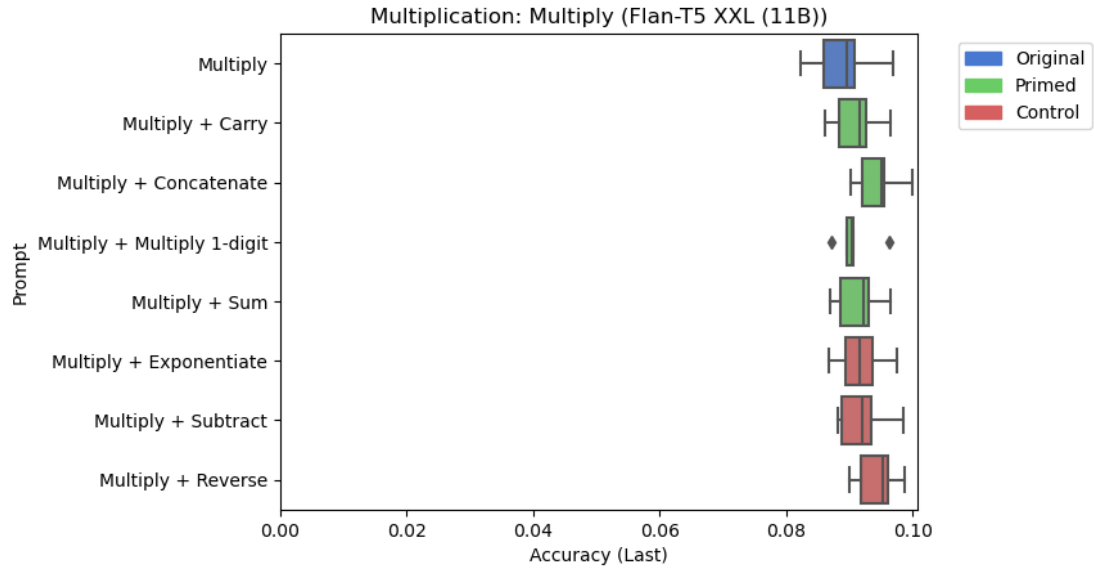
8b: Flan-T5 Base's accuracy for multiplication using different prompts.



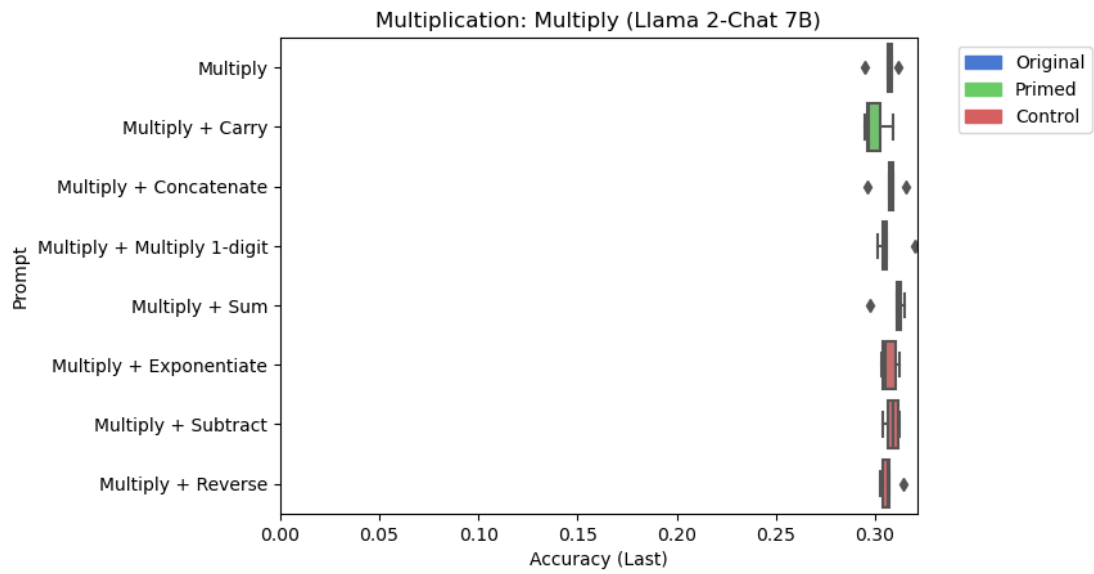
8c: Flan-T5 Large's accuracy for multiplication using different prompts.



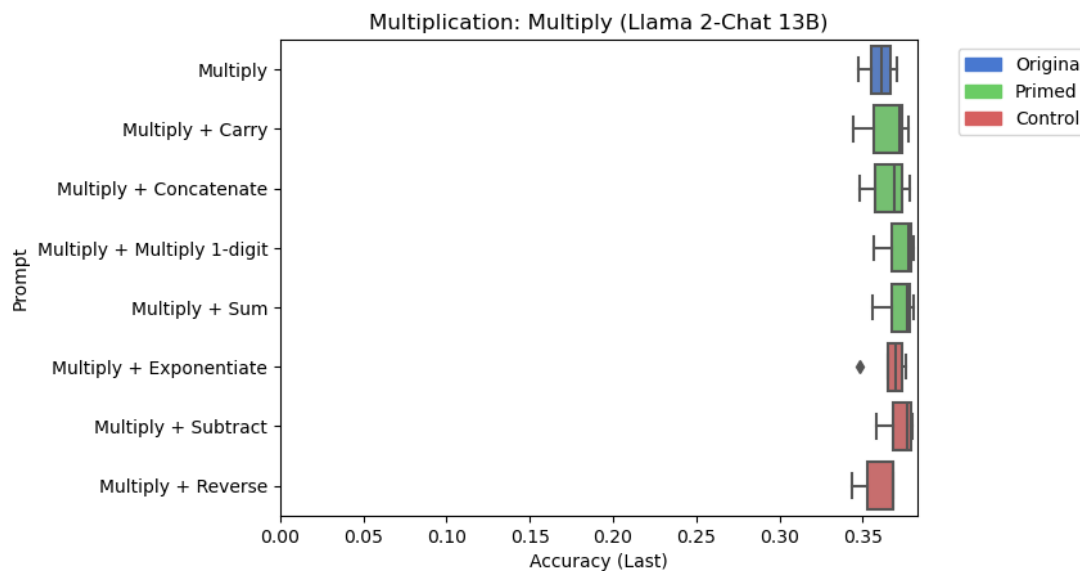
8d: Flan-T5 XL's accuracy for multiplication using different prompts.



8e: Flan-T5 XXL's accuracy for multiplication using different prompts.



8f: Llama 2-Chat 7B's accuracy for multiplication using different prompts.



8g: Llama 2-Chat 13B's accuracy for multiplication using different prompts.

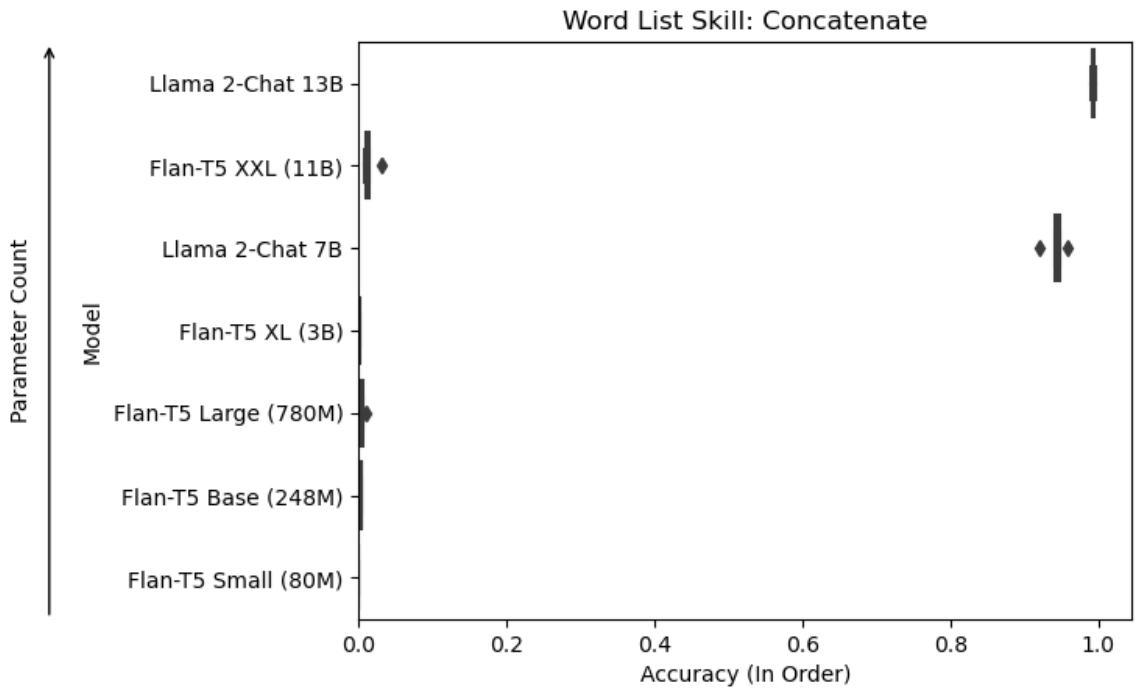
Figure 8: Results for the multiplication task using different models.

There are a few things to note from the results in Figure 8. Llama-2 Chat 13B performed the best at multiplication, but its accuracy was still poor at around 35%. While the Multiply + Concatenate prompt seemed to improve the response for Flan-T5 Base, the overall accuracy is still poor at around 1.3%. Furthermore, none of the other models responded similarly when primed with related skills. Including the control tasks for other models had inconsistent effects on model performance. Flan-T5 XL performed slightly worse when including the string reversal control task to the prompt on average, but Flan-T5 Large performed marginally better with string reversal than other prompts on average. However, these differences are negligible due to these models' trivial performances. For larger models, the control task did not significantly affect model performance.

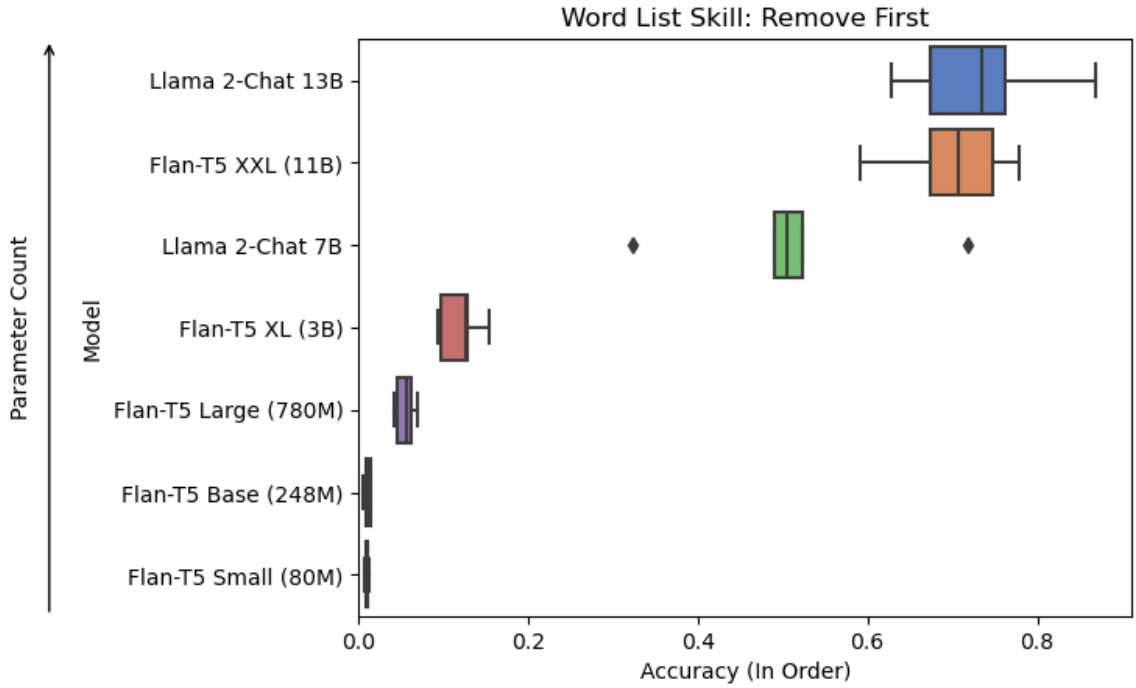
WORD LIST

For the word list task, the models were evaluated on the following skills: concatenation, remove first word, and uppercase all words. Furthermore, the models were evaluated on a word list operation (WLO), with and without being primed on a skill.

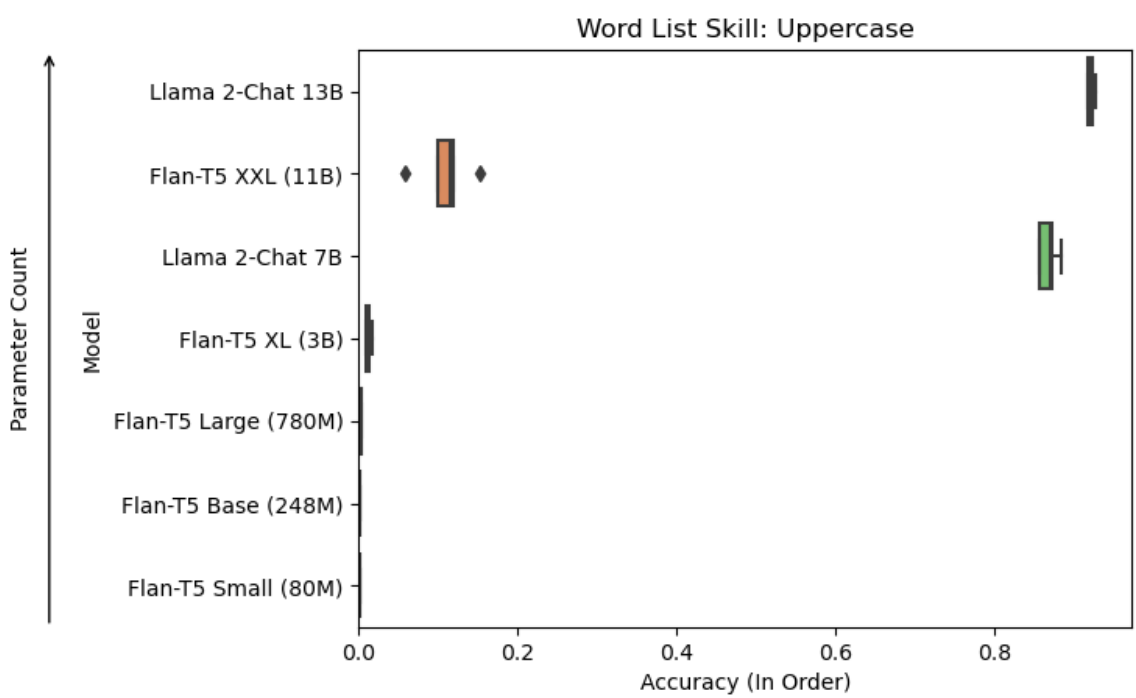
Figure 9 shows the results of the skills across all models used in the experiment. Figure 10 shows the results of the main task across different prompts for the Llama 2-Chat models only due to the poor results from Flan-T5 for this experiment.



9a: Models of increasing size evaluated on accuracy for the concatenation skill.



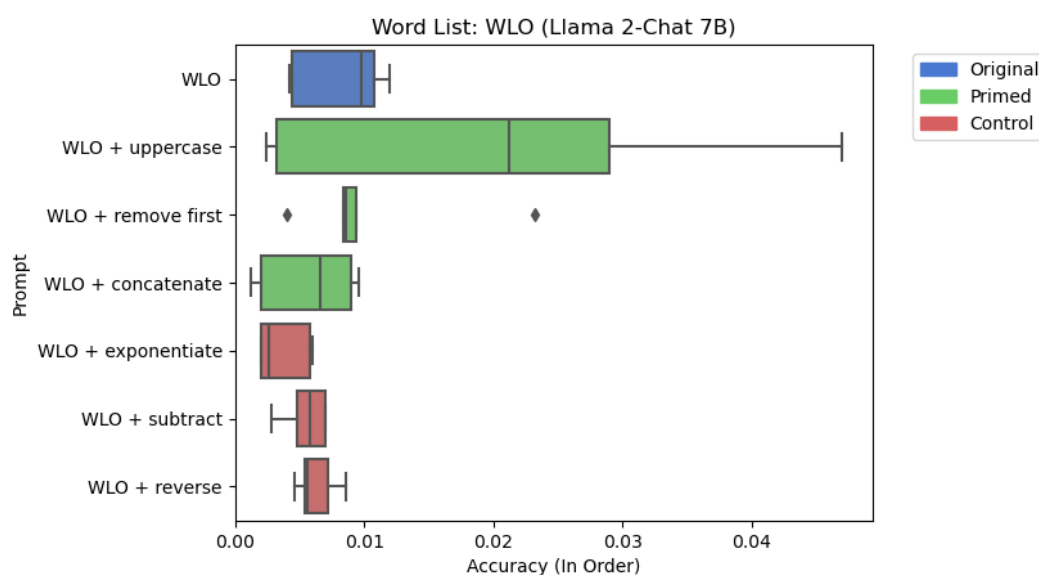
9b: Models of increasing size evaluated on accuracy for the remove first skill.



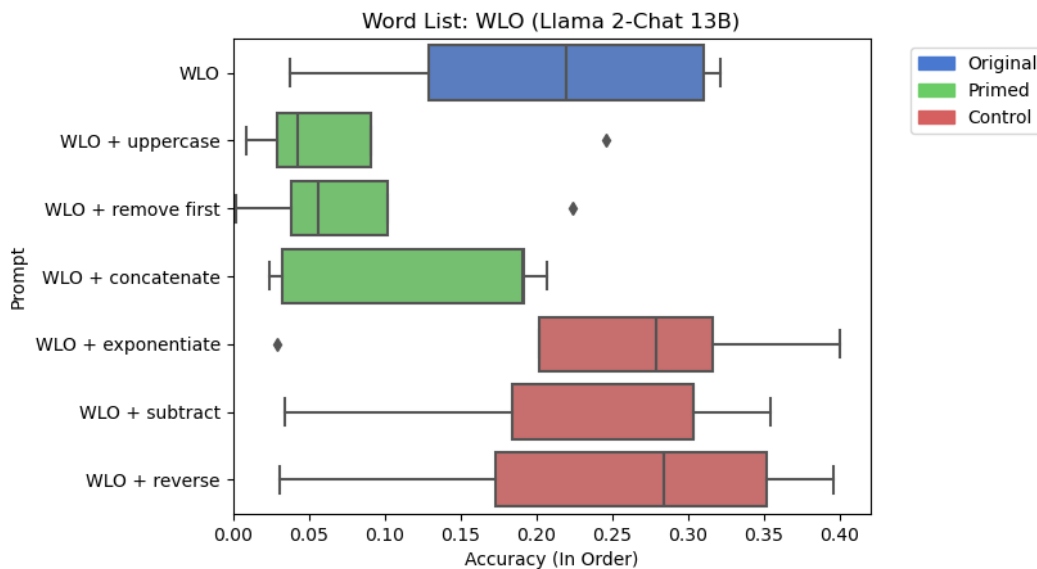
9c: Models of increasing size evaluated on accuracy for the uppercase skill.

Figure 9: Results for skills used in WLO.

For the skills needed to perform WLO, only the Llama 2-Chat models could perform them reliably. Of the Flan-T5 models, Flan-T5 XXL performed the best in the skills overall. It performed on par with the Llama 2-Chat models in the remove first skill but poorly on the concatenation and uppercase skills. Upon further inspection, the Flan-T5 models tended not to change the word list and mistake uppercasing all words as uppercasing the first letter in each word. For the concatenation task, it tended to concatenate all words into a single word. These results contrast with Llama 2-Chat, which was able to concatenate the lists into a comma-delimited word list.



10a: Llama 2-Chat 7B's accuracy for word list operation using different prompts.



10b: Llama 2-Chat 13B's accuracy for word list operation using different prompts.

Figure 10: Results for the WLO task using different models.

For Llama 2-Chat 7B, the introduction of the uppercase skill did improve some of the runs. However, the accuracies are still negligible. Looking at the larger model, Llama 2-Chat 13B did not exhibit this behavior and tends to perform better with control prompts on average. However, the accuracies between each in-context example varied substantially, highlighting the importance of selecting the right one for a task.

Upon further examination, Llama 2-Chat models tended to misunderstand how concatenation operated in the WLO experiment. Instead of concatenating the two lists into one list, they sometimes combined the lists into a single word, much like Flan-T5 did with the concatenation task. Even with the injection of concatenation skills, the models did not pick up on this distinction. Sometimes, the models mistook which words belonged to which list, capitalized all words in both lists, and did not remove the first word in the second list.

DISCUSSION & CONCLUSION

Previous authors had performed work to understand the behavior large language models exhibit when reasoning with compositional tasks. Dziri et al. determine the limits of compositionality in transformer-based models based on reasoning with computational graphs based on subtask operations [2]. Despite prompting models to decompose a compositional task, they found that these models tended to fail as the depth of the graph computational graph increased. Along similar lines, Press et al. investigated the relationship between model size and its ability to reason in compositional tasks [18]. They found that the compositional gap between single-hop and multi-hop question-answering does not decrease as model sizes increase. Lepori et al. highlight one way neural networks can reason with complex tasks is because there exists structural subcomponents of a network that can handle subtasks [19]. They created models trained on odd-one-out language tasks and found that one can remove a subnetwork that handles a subtask from a network and leave the network’s functionality of other subtasks intact.

Furthermore, previous authors have explored numerical reasoning in large language models. One of Dziri et al.’s experiments involved multiplication, and they divided it into five subtasks: “one-digit multiplication, sum, mod 10, carry over, [and] concatenation” [2]. They used the number of digits between the two numbers multiplied to represent the size of the problem. Ling et al. created a framework for LLMs to solve algebraic problems by “generating answer rationales” alongside the predicted answer [20]. Answer rationales create an interpretable explanation of why a model chose a particular answer. Ontañón et al. explored different design patterns for the transformer architecture for compositional tasks, including “addition, duplication, [and] set

intersection” [21]. They found that improvements from architectural changes depended on the compositional task evaluated.

In the multiplication experiment, we observed a trend of large models performing better than smaller models. We can also observe a similar effect with the Llama-2 Chat models in the WLO experiment. Our results also prove that models cannot learn compositionality in-context reliably. Even if a model can perform relevant skills well individually, including an in-context example of relevant skills did not consistently improve the model. In the larger models for multiplication, including a relevant or irrelevant skill did not change the model's overall performance. For the WLO experiment, a relevant or irrelevant skill had inconsistent effects on model performance.

Several limitations of these experiments revolve around scaling and resource constraints. The largest model considered for these experiments was Llama 2-Chat 13B, but the largest Llama 2-Chat model available is 70B. Only the Flan-T5 and Llama 2-Chat family of models were considered, and future experimentation should consider different families, such as GPT. The choice of how prompts are given to a model can influence model performance [5], but only one main prompt format was considered for these experiments. Furthermore, future work could add more in-context examples and combine skill examples into a single prompt. We considered in-context examples for our experiments, but future work can try finetuning models on the subtasks instead. We conducted experiments involving multiplication and word list operation. A natural extension is to design additional tasks to see if our observations hold across them.

REFERENCES

- [1] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, H. Nori, H. Palangi, M. T. Ribeiro, and Y. Zhang, “Sparks of Artificial General Intelligence: Early experiments with GPT-4.” arXiv, Apr. 13, 2023. doi: 10.48550/arXiv.2303.12712.
- [2] N. Dziri, X. Lu, M. Sclar, X. L. Li, L. Jiang, B. Y. Lin, P. West, C. Bhagavatula, R. L. Bras, J. D. Hwang, S. Sanyal, S. Welleck, X. Ren, A. Ettinger, Z. Harchaoui, and Y. Choi, “Faith and Fate: Limits of Transformers on Compositionality.” arXiv, Jun. 01, 2023. doi: 10.48550/arXiv.2305.18654.
- [3] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S. S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, A. Castro-Ros, M. Pellat, K. Robinson, D. Valter, S. Narang, G. Mishra, A. Yu, V. Zhao, Y. Huang, A. Dai, H. Yu, S. Petrov, E. H. Chi, J. Dean, J. Devlin, A. Roberts, D. Zhou, Q. V. Le, and J. Wei, “Scaling Instruction-Finetuned Language Models,” *Journal of Machine Learning Research*, vol. 25, no. 70, pp. 1–53, 2024.
- [4] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith,

- R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom, “Llama 2: Open Foundation and Fine-Tuned Chat Models.” arXiv, Jul. 19, 2023. Accessed: Mar. 23, 2024. [Online]. Available: <http://arxiv.org/abs/2307.09288>
- [5] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, “Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing,” *ACM Comput. Surv.*, vol. 55, no. 9, p. 195:1-195:35, Jan. 2023, doi: 10.1145/3560815.
- [6] T. Kuribayashi, Y. Oseki, T. Ito, R. Yoshida, M. Asahara, and K. Inui, “Lower Perplexity is Not Always Human-Like,” in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Online: Association for Computational Linguistics, 2021, pp. 5203–5217. doi: 10.18653/v1/2021.acl-long.405.
- [7] D. Ippolito, R. Kriz, J. Sedoc, M. Kustikova, and C. Callison-Burch, “Comparison of Diverse Decoding Methods from Conditional Language Models,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, A. Korhonen, D. Traum, and L. Márquez, Eds., Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 3752–3762. doi: 10.18653/v1/P19-1365.
- [8] S. Zhang, L. Dong, X. Li, S. Zhang, X. Sun, S. Wang, J. Li, R. Hu, T. Zhang, F. Wu, and G. Wang, “Instruction Tuning for Large Language Models: A Survey.” arXiv, Oct. 09, 2023. doi: 10.48550/arXiv.2308.10792.

- [9] S. Min, X. Lyu, A. Holtzman, M. Artetxe, M. Lewis, H. Hajishirzi, and L. 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*, Y. Goldberg, Z. Kozareva, and Y. Zhang, Eds., Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 11048–11064. doi: 10.18653/v1/2022.emnlp-main.759.
- [10] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, “Language Models are Few-Shot Learners,” in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2020, pp. 1877–1901. Accessed: Apr. 07, 2024. [Online]. Available: <https://papers.nips.cc/paper/2020/hash/1457c0d6bfc4967418bfb8ac142f64a-Abstract.html>
- [11] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, “Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank,” in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, D. Yarowsky, T. Baldwin, A. Korhonen, K. Livescu, and S. Bethard, Eds., Seattle, Washington, USA: Association for Computational Linguistics, Oct. 2013, pp. 1631–1642. Accessed: Mar. 26, 2024. [Online]. Available: <https://aclanthology.org/D13-1170>

- [12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. ukasz Kaiser, and I. Polosukhin, “Attention is All you Need,” in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2017. Accessed: Apr. 11, 2024. [Online]. Available: https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html
- [13] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *J. Mach. Learn. Res.*, vol. 21, no. 1, p. 140:5485-140:5551, Jan. 2020.
- [14] Z. G. Szabó, “Compositionality,” in *The Stanford Encyclopedia of Philosophy*, Fall 2022., E. N. Zalta and U. Nodelman, Eds., Metaphysics Research Lab, Stanford University, 2022. Accessed: Mar. 30, 2024. [Online]. Available: <https://plato.stanford.edu/archives/fall2022/entries/compositionality/>
- [15] T. M. V. Janssen and B. H. Partee, “Chapter 7 - Compositionality,” in *Handbook of Logic and Language*, J. van Benthem and A. ter Meulen, Eds., Amsterdam: North-Holland, 1997, pp. 417–473. doi: 10.1016/B978-044481714-3/50011-4.
- [16] “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models.” Accessed: Nov. 08, 2023. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html
- [17] J. Wei, X. Wang, D. Schuurmans, M. Bosma, brian ichter, F. Xia, E. Chi, Q. V. Le, and D. Zhou, “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models,” in *Advances in Neural Information Processing Systems*, S. Koyejo, S.

- Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., Curran Associates, Inc., 2022, pp. 24824–24837. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf
- [18] O. Press, M. Zhang, S. Min, L. Schmidt, N. A. Smith, and M. Lewis, “Measuring and Narrowing the Compositionality Gap in Language Models.” May 22, 2023. doi: 10.48550/arXiv.2210.03350.
- [19] M. A. Lepori, T. Serre, and E. Pavlick, “Break It Down: Evidence for Structural Compositionality in Neural Networks.” arXiv, Jan. 25, 2023. doi: 10.48550/arXiv.2301.10884.
- [20] W. Ling, D. Yogatama, C. Dyer, and P. Blunsom, “Program Induction by Rationale Generation: Learning to Solve and Explain Algebraic Word Problems,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, R. Barzilay and M.-Y. Kan, Eds., Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 158–167. doi: 10.18653/v1/P17-1015.
- [21] S. Ontanon, J. Ainslie, Z. Fisher, and V. Cvicek, “Making Transformers Solve Compositional Tasks,” in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, S. Muresan, P. Nakov, and A. Villavicencio, Eds., Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 3591–3607. doi: 10.18653/v1/2022.acl-long.251.

Name of Candidate: Jordan Tan

Date of Submission: May 6, 2024