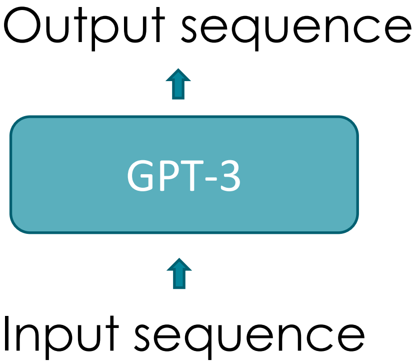
Limitations of Language Models in Arithmetic Induction

Xifeng Yan
with Jing Qian and Hong Wang

Computer Science Department
University of California at Santa Barbara

Jing et al. Limitations of Language Models in Arithmetic and Symbolic Induction

Large Language Models



Pretraining tasks
+
Large Transformer model

Large text corpus

GPT-3 Success

Text in red is generated by the model

Translate this into 1. French, 2. Spanish and 3. Chinese:

What rooms do you have available?

1. Quels sont les chambres disponibles?

2. Cuáles son las habitaciones disponibles?

3. 您有什么房间可供选择?

Zero-shot
machine translation

Q: How does a telescope work?

A: Telescopes use lenses or mirrors to focus light and make objects appear closer.

Q: Where were the 1992 Olympics held?

A: The 1992 Olympics were held in Barcelona, Spain.

Q: Where is the Valley of Kings?

A: The Valley of Kings is located in Luxor, Egypt.

Few-shot
question answering

My Experience

Kid: I can't get medical service because I'm in Siberia

Bot: Sorry to hear that you're having trouble getting medical service. if you're in siberia, there may not be many options for medical care. however, you could try looking into online resources or contacting your local embassy for more information.

Scaling

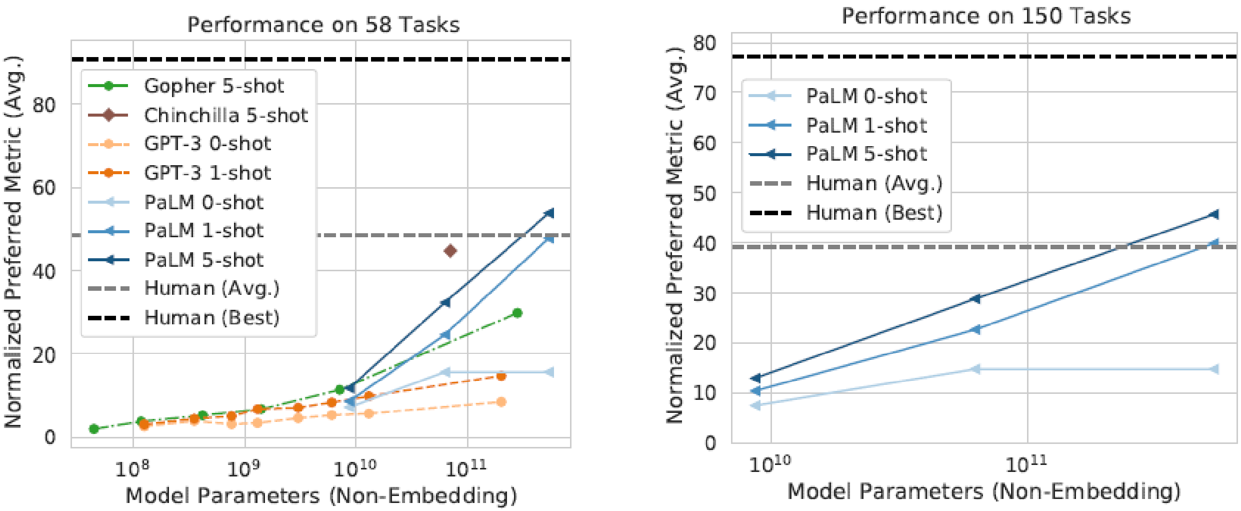


Figure 3: BIG-bench evaluation of PaLM. (left) Evaluation of PaLM, GPT-3, Gopher, and Chinchilla. Previous models have only evaluated on a subset of tasks, so this graph shows aggregate results on the 58 tasks which all three models have evaluated on. (right) Evaluation of PaLM on a larger set of 150 BIG-bench tasks. For each task, the results of its preferred metric are used. The results are normalized by setting the maximum score to 100 and the random chance score to 0 for multiple-choice tasks, so that they are negative valued if the model performs worse than random chance. The normalized results are averaged over all tasks.

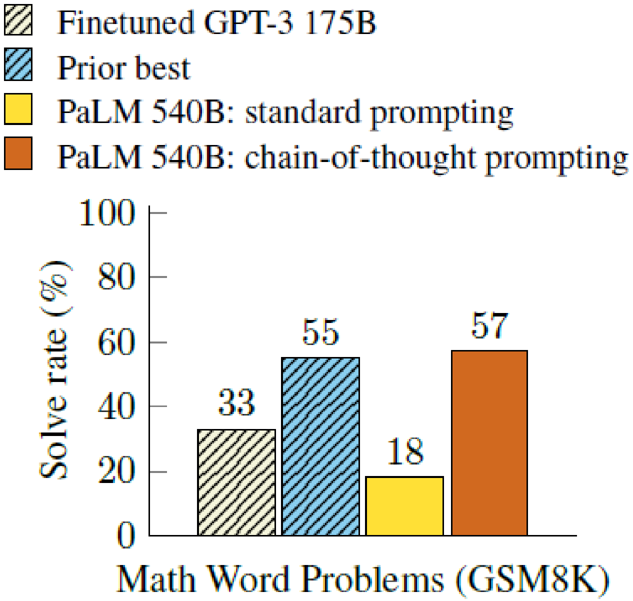
Chowdhery et al. PaLM: Scaling Language Modeling with Pathways

Further Improvement: Chain-of-Thought Prompting

Math Word Problems (free res

Q: Roger has 5 tennis balls. H
2 more cans of tennis balls. E
has 3 tennis balls. How many
balls does he have now?

A: Roger started with 5 balls.
of 3 tennis balls each is 6 ten
balls. 5 + 6 = 11. The answer



CSQA (commonsense)

Q: Sammy wanted to go to where the people were. Where might he go?
Options: (a) race track (b) populated areas (c) desert (d) apartment (e) roadblock

A: The answer must be a place with a lot of people. Race tracks, desert, apartments, and roadblocks don't have a lot of people, but populated areas do. So the answer is (b).

Wei et al. Chain of thought prompting elicits reasoning in large language models.

Observation

Observation:

Larger Models => Better Results
More details => Better Results

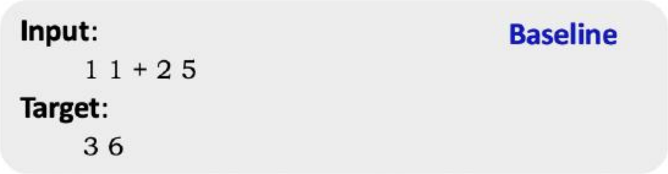
Question:

Is their goal to achieve **100%** accuracy?

If Yes, is it really possible?
If Not, what is the goal?

Arithmetic Induction

Scratchpad: show intermediate computation steps



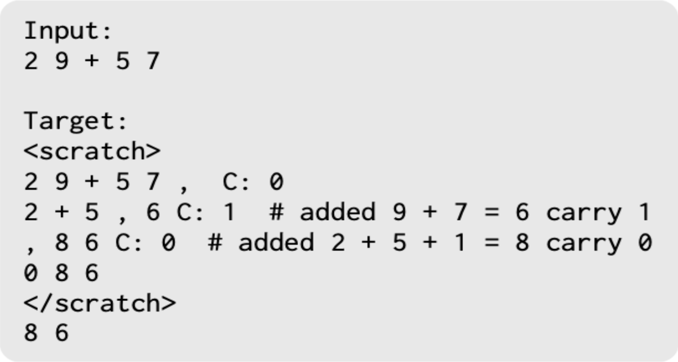
Input:

1 1 + 2 5

Target:

3 6

Baseline



```
Input:
2 9 + 5 7

Target:
<scratch>
2 9 + 5 7 ,  C: 0
2 + 5 , 6 C: 1  # added 9 + 7 = 6 carry 1
, 8 6 C: 0  # added 2 + 5 + 1 = 8 carry 0
0 8 6
</scratch>
8 6
```

Nye, et al. Show your work: Scratchpads for intermediate computation with language models

Arithmetic Induction

Scratchpad: show intermediate computation steps

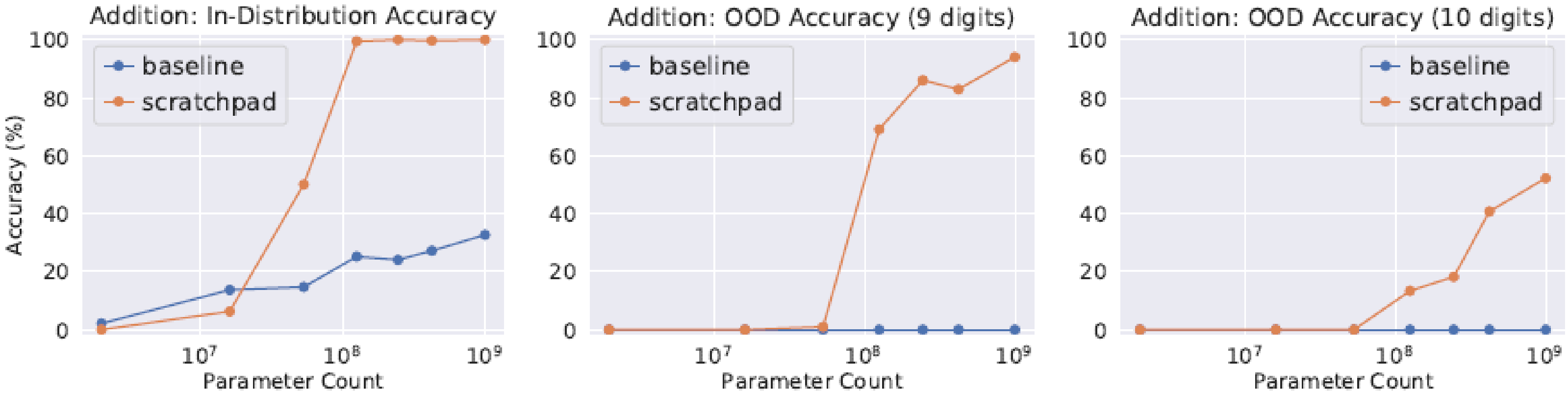


Figure 3: Using a scratchpad significantly improves the performance of pre-trained Transformer-based models on addition, including their ability to generalize out of the training distribution to numbers with more digits. Models were trained on 1-8 digit addition. The baseline models were trained without intermediate scratchpad steps.

Nye, et al. Show your work: Scratchpads for intermediate computation with language models

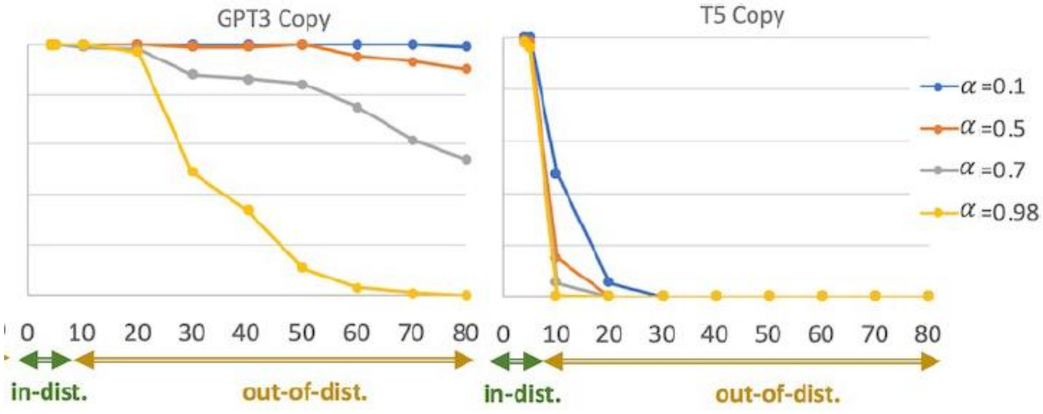
Other Simple Symbol Manipulation Tasks: Copy and Reverse

n-digit copy (input: 9 3 3 1 5 output: 9 3 3 1 5)

n-digit reverse (input: 9 3 3 1 5 output: 5 1 3 3 9)

It Cannot Generalize!

n-digit copy (input: 9 3 3 1 5 output: 9 3 3 1 5)



An example x_1, … x_n with n digits are sampled with the next digit probability $p(x_{i+1} \mid x_i) = \alpha$, when $x_{i+1} = x_i$; otherwise, $(1 - \alpha)/9$. Larger indicates a higher repetition level.

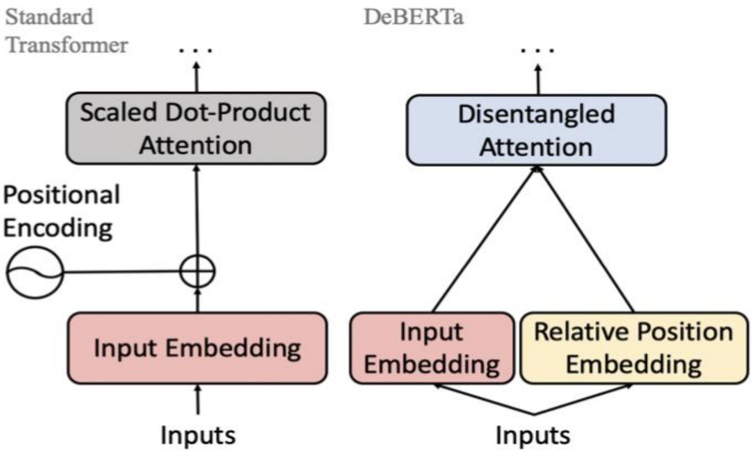
Jing et al. Limitations of Language Models in Arithmetic and Symbolic Induction

Two Issues Exist

- The model needs to figure out where each digit comes from, especially with repeating digits.

- The model can’t generalize to OOD cases, e.g., 10-digit addition when only observing addition on 1-9 digits.

Is it caused by embedding?



Use position marker to differentiate the repeating tokens:

- The attention scores between tokens depend not only on the content but also on the relative position between the tokens.
- The disentangled relative position embeddings act as implicit position markers within DeBERTa.

He et al., DeBERTa: Decoding-enhanced BERT with Disentangled Attention

Explicit position marker: marker tokens

For each digit in the number, add a unique token as marker before it

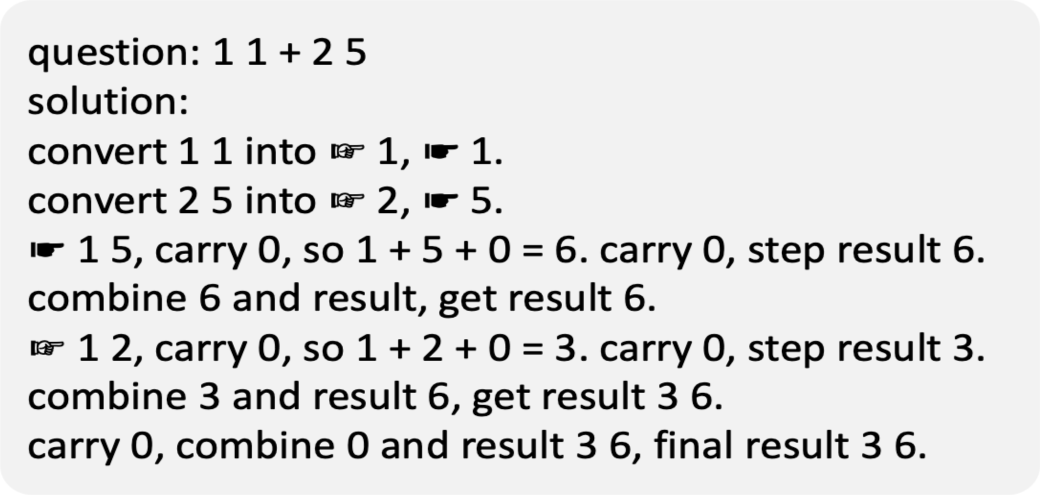
Ordered marker: the markers are inserted into the input in order
Example: 2 2 2→A 2 B 2 C 2

Random marker: the markers are inserted into the input in random order
Example: 2 2 2→E 2 X 2 J 2

What is good about explicit position marker:

- It breaks the repeating numbers into a non-repeating input sequence
- It provide more direct and clearer location information in text format

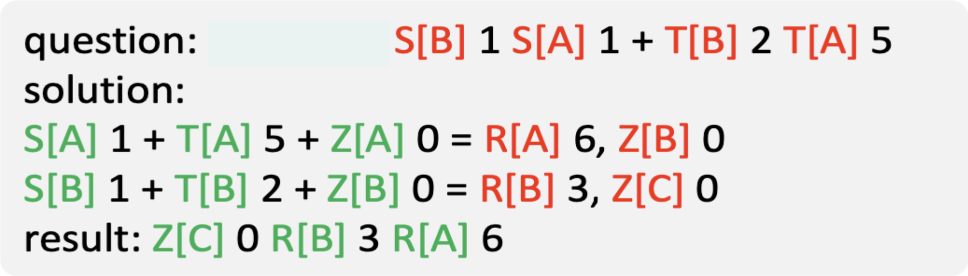
Add position marker to scratchpad



question: 1 1 + 2 5
solution:
convert 1 1 into ☞ 1, ☛ 1.
convert 2 5 into ☞ 2, ☛ 5.
☛ 1 5, carry 0, so 1 + 5 + 0 = 6. carry 0, step result 6.
combine 6 and result, get result 6.
☞ 1 2, carry 0, so 1 + 2 + 0 = 3. carry 0, step result 3.
combine 3 and result 6, get result 3 6.
carry 0, combine 0 and result 3 6, final result 3 6.

☞ and ☛ are optional position markers.

Reference Marker



question: S[B] 1 S[A] 1 + T[B] 2 T[A] 5

solution:

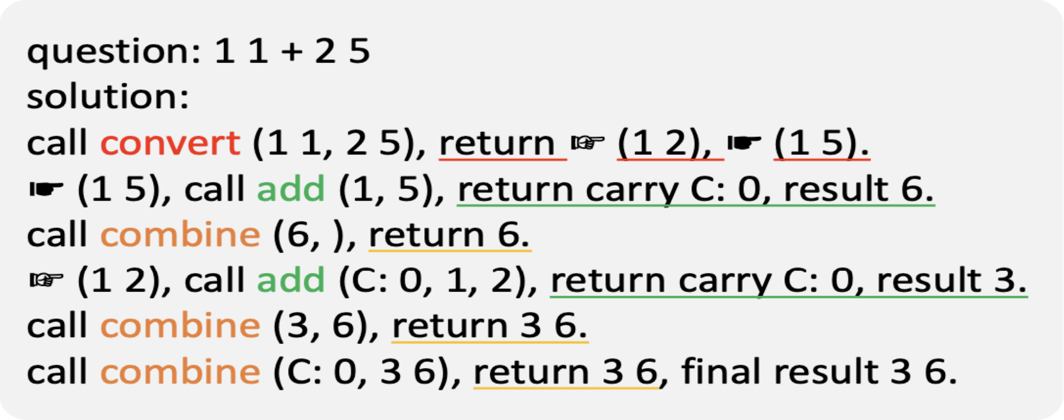
S[A] 1 + T[A] 5 + Z[A] 0 = R[A] 6, Z[B] 0

S[B] 1 + T[B] 2 + Z[B] 0 = R[B] 3, Z[C] 0

result: Z[C] 0 R[B] 3 R[A] 6

- The position marker (first appearance) are for the new defined variables. Each position marker will a value associated with it.
- The reference pointer (later appearances) are referring to the previous position marker where we will copy the value from.
- The model only needs to correctly generate the reference marker when it want to reuse previous number.

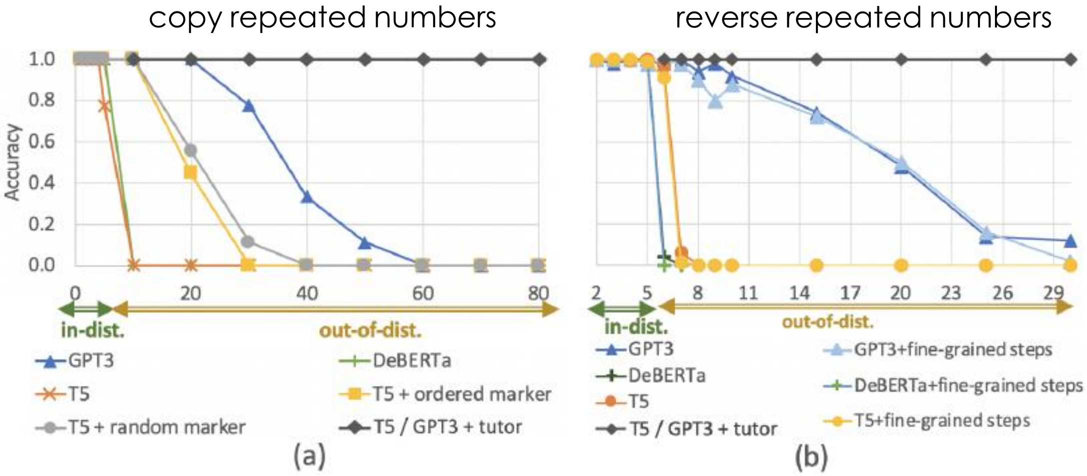
Callable Program



question: 1 1 + 2 5
solution:
call convert (1 1, 2 5), return ☞ (1 2), ☛ (1 5).
☛ (1 5), call add (1, 5), return carry C: 0, result 6.
call combine (6,), return 6.
☞ (1 2), call add (C: 0, 1, 2), return carry C: 0, result 3.
call combine (3, 6), return 3 6.
call combine (C: 0, 3 6), return 3 6, final result 3 6.

- Define the programs that can solve sub-problems such as adding one digits.

With All These Modifications



copy repeated numbers

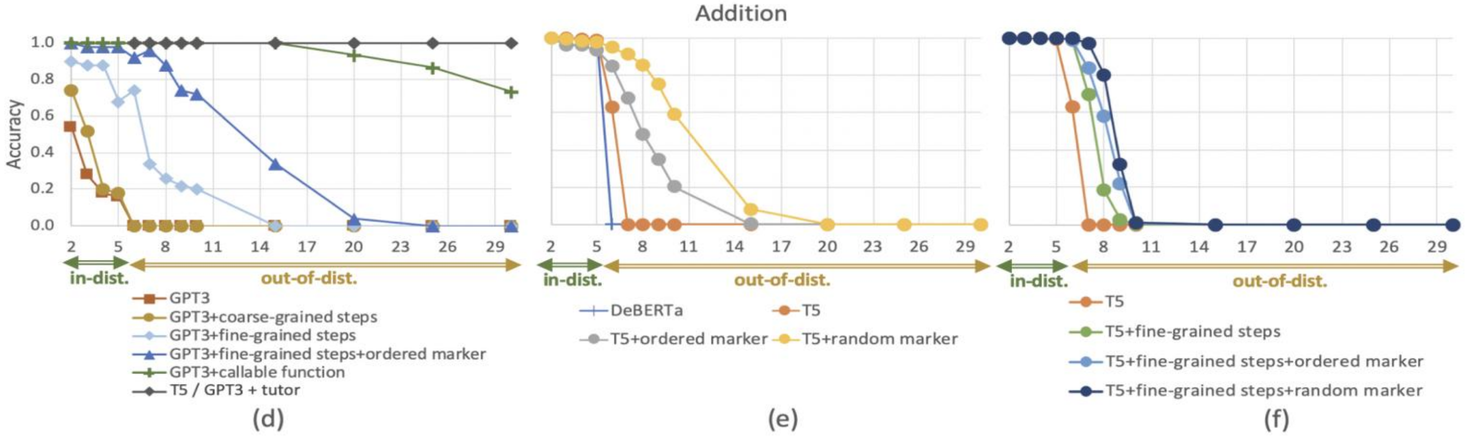
reverse repeated numbers

(a)

(b)

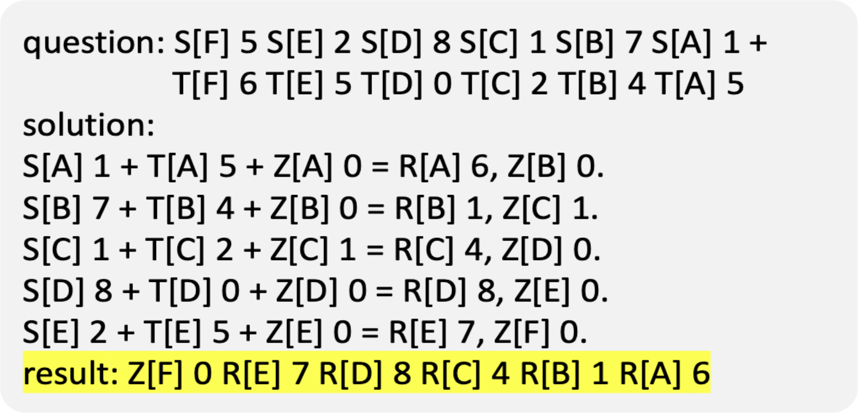
- For copy task, GPT3 achieves 100% accuracy on the in-distribution testing data (1-5 digits); the fine-tuned T5 achieves 78% accuracy.
- Augmented with random or ordered positional markers, the T5 models achieve 100% in-distribution accuracy, and so does DeBERTa.
- GPT3 exhibits better OOD generalization than T5 with positional markers but it does not generalize well beyond 30 digits.

Results on Addition



- More fine-grained steps work better on GPT-3.
- The performance of GPT-3 is further boosted with explicit positional markers.
- Explicit positional markers might make it easier for LMs to learn the induction in the arithmetic reasoning tasks.

Can We Handle OOD Now?



```
question: S[F] 5 S[E] 2 S[D] 8 S[C] 1 S[B] 7 S[A] 1 +
          T[F] 6 T[E] 5 T[D] 0 T[C] 2 T[B] 4 T[A] 5
solution:
S[A] 1 + T[A] 5 + Z[A] 0 = R[A] 6, Z[B] 0.
S[B] 7 + T[B] 4 + Z[B] 0 = R[B] 1, Z[C] 1.
S[C] 1 + T[C] 2 + Z[C] 1 = R[C] 4, Z[D] 0.
S[D] 8 + T[D] 0 + Z[D] 0 = R[D] 8, Z[E] 0.
S[E] 2 + T[E] 5 + Z[E] 0 = R[E] 7, Z[F] 0.
result: Z[F] 0 R[E] 7 R[D] 8 R[C] 4 R[B] 1 R[A] 6
```

Not yet:

- The LM tends to overfit training data. The model cannot generalize to 6-digit addition, if only trained on 5 digits
- The model may learn some complicated computational rules to associate the tokens in the training data. There is no guarantee that the model can find the underlying mechanisms for addition

What is the Challenge?

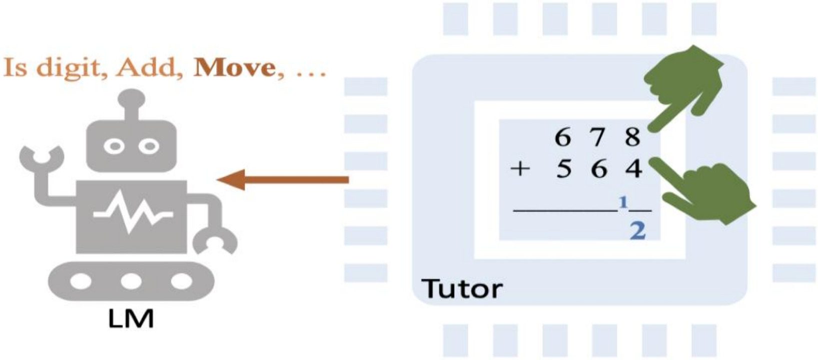
Generalization from n to n+1

Rollback to the Origin: How Does a Kid Learn Addition?

$$\begin{array}{r} 786 \\ +\,467 \\ \hline \end{array}$$

- The teacher will point out which numbers are used at each step.
- Different actions to manipulate the number such as add, carry on.
- Have a termination rule when we should stop the process.

Language Models with Tutor

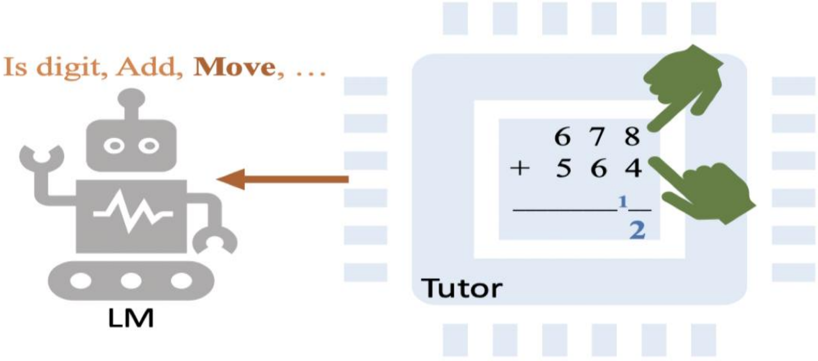


- Imitate the addition action sequence that a human does
- This setting is like a Turing machine, where LM learns the state transition
- We further assume all individual actions have been learned

We limited the operation space

Example of an action sequence for the addition task

- “**lmov**” denotes moving the cursor one token to the left.
- “**add**” adds three single digits
- “**end**” checks if there is any digit left
  - “end=F” there are some digits
  - “end=T” no digit left



lmov, end=F, add, lmov, end=F, add, . . . , lmov, end=T

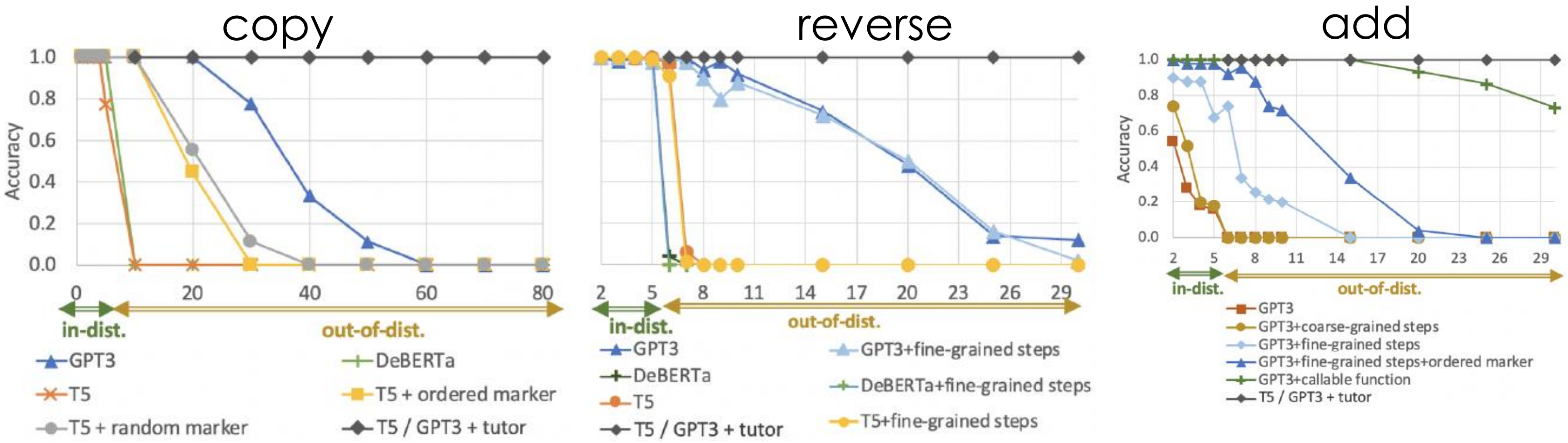
A transformer generates an action sequence that simulates how humans do arithmetic addition

Example of an action sequence for the copy task

- “**rmov**” denotes moving the cursor one token to the right
- “**cpy**” copies a single digit to the output sequence
- “**end**” checks if there is any digit left
  - “end=F”
  - “end=T”

rmov, end=F, cpy, rmov, end=F, cpy, . . . , rmov, end=T

Results on addition problem



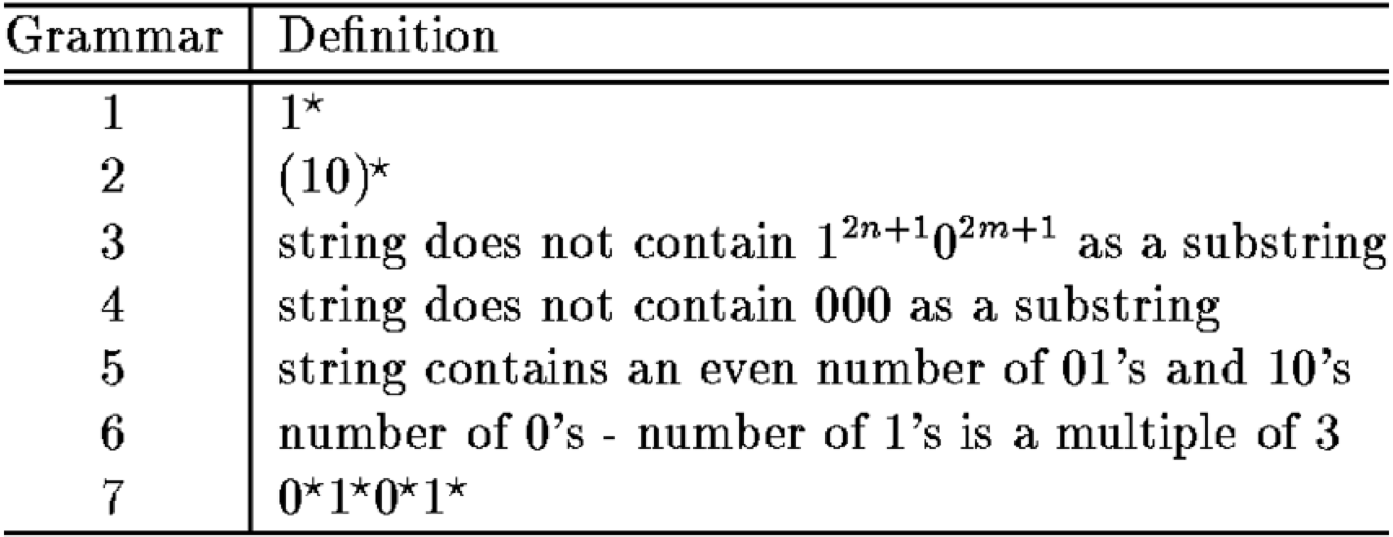
- *Now better*
- *LMs + tutor* keeps 100% accuracy on all the experimented numbers of digits

Can this method shed some light on other symbol induction tasks?

Can this method be generalized?

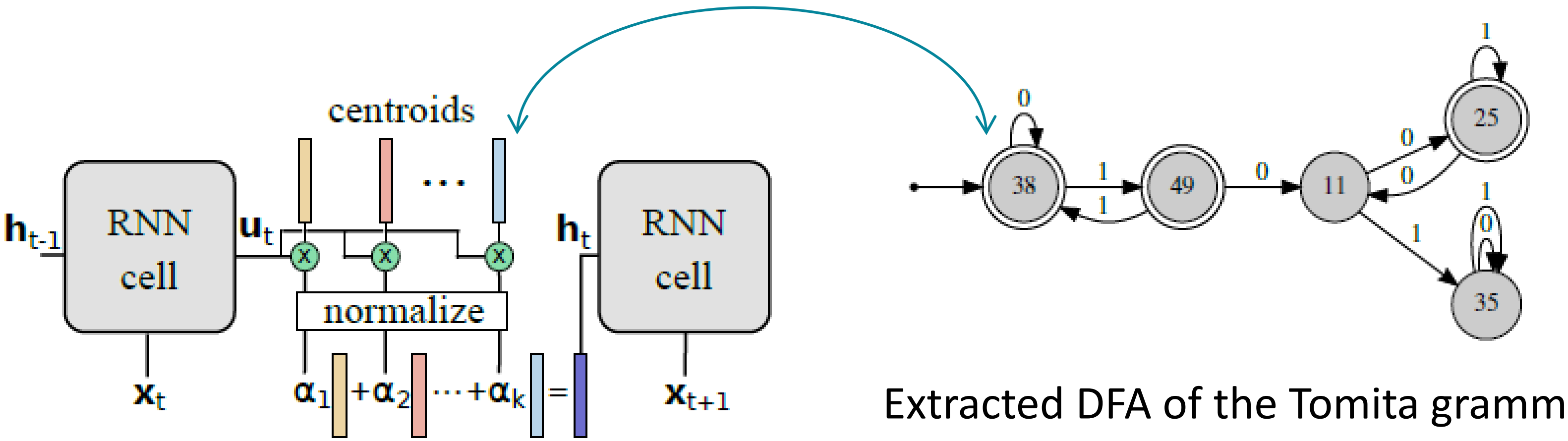
The action sequence shares similarity with a Turing machine that operates on a tape.

Tomita grammars



Grammar	Definition
1	$1^\star$
2	$(10)^\star$
3	string does not contain $1^{2n+1}0^{2m+1}$ as a substring
4	string does not contain 000 as a substring
5	string contains an even number of 01's and 10's
6	number of 0's - number of 1's is a multiple of 3
7	$0^\star 1^\star 0^\star 1^\star$

State-Regularized Recurrent Neural Networks



Extracted DFA of the Tomita grammars

Wang and Niepert, State-Regularized Recurrent Neural Networks, ICML 2019

It might not be enough

as we were spoiled by GPT-3 !

1. Does not work if there are only a few demonstrations

2. Has to be retrained for individual tasks

3. Does not work if there is no step-by-step demonstration

Take-Away Messages

- The performance of LLM will improve to solve symbolic manipulation using fine-grained demonstrate steps. They fail on the situations of repeated symbols and OOD
- To alleviate the issue of repeating symbols, we can use position markers, more fine-grained prompts, reference markers and even callable programs. However, it still fail on OOD
- We use LM with tutor to learn the action sequence, which has the potential of solving OOD

- More work needs to be done if the goal is to achieve 100% accuracy
- Purely increasing the model size seems not enough

Q & A