Investigating Formal Chain of Reasoning for NumglueTests

Group Name: Dolin-Vora
Pavel Dolin (ASU ID: 1225393325)
Aakash Sudhirbhai Vora (ASU ID: 1225441698)

Project Mentors: Mihir Parmar & Neeraj Varshney

1. Introduction

Word problems involving arithmetic and, in general, numerical reasoning are still challenging problems in AI research. Various approaches have been taken to tackle these problems. However, the two broad categories outlined in the previous works are Symbolic approaches and End-to-end Language Models. Traditional NLP systems have shown that Symbolic approaches are robust but less powerful. Whereas it has been shown by (Mishra et al., 2022) that the end-to-end approach for GPT3 performs extremely poorly. We are focusing on combining the best of both worlds in the project. We test out Few-Shot techniques, or prompt engineering techniques, and present a novel approach, which we call, Formal Chain of Reasoning. Our method outperforms the best baselines and demonstrates the potential for general use for other tasks. Furthermore, we present an analysis of the performance of our techniques on a range of available GPT3 models like davinci, ada, curie, etc. For the latter part of the study, we present how we trained GPT2, GPT1, and T5 to obtain results comparable to some GPT3 models.

2. Methods

Dataset

To bridge the gap and bring the models to human-level performance (Mishra et al., 2022), identify eight main challenges and define a benchmark - NumGLUE. It consists of ~100k questions combining logical and numerical reasoning tasks that require arithmetic understanding, common sense, domain-specific knowledge, interpolation, and extrapolation skills. From the total of eight tasks, we are only considering a subset of tasks from the total of eight tasks for our experiment.

Synthetic dataset generation

NumGLUE contains ~1000 training data points which can be very few when fine-tuning a large model like GPT2 with 117M parameters. Thus, we sensed the need to generate more data so the model could train on enough data. We applied several transformations on the training dataset for TASK 4 from NumGLUE and generated around 0.4M (414,505) data points. For the data points available in TASK 4, we used GPT3 to generate labels and applied transformations on the points labeled correctly. Structural transformation, Name transformation, and Numerical transformation were considered in our experiments.

Structural Transformation

For changing the structure of the input sentence without changing the meaning, we used two models, a pretrained T5 model (translation_en_to_de) for English to German translation and a pretrained Bert2Bert model (google/bert2bert_L-24_wmt_de_en) for translating sentences in German back to English. For example,

Sentence: Edward was selling his old games. He started out with 35 but sold 19 of them. He packed the rest up putting 8 games into each box. He had to use _____ boxes.
Post transformation: Edward sold his old games. He began his career at the age of 35, selling 19 of them. He packed the rest and put 8 games in each box. He had to use _____ - boxes.

Name Transformation

Ideally; we can create a model that can extract equations from the sentences; it should be indifferent to changes to the names of entities. To enforce this, we also introduced a transformation that can take a sentence and replace the names of all human entities with new ones. This was a complicated transformation since it would require us to handle all the pronouns related to the name we are changing. We divided this task into two parts. First, identifying persons in the sentence and second, linking the identified person with the pronouns referring to them. For the first part, we used the Named Entity Recognition model from scipy, and for the second part, we used the neuralcoref model from huggingface.

For example,

Sentence: Mariela was in the hospital, and she got 403 get well cards from around the country. When she got home, she got 287 more cards from friends and family. Mariela got _____ get well cards.
Post transformation: Burnell was in the hospital, and he got 53 get well cards from around the country. When he got home he got 428 more cards from friends and family. Burnell got _____ get well cards.

Numerical Transformation

Finally, we wanted to ensure the model's performance should be independent of the size of the numbers in the input and enforce generalization to numbers outside of the training data. For this, we took each input sentence from the training dataset and replaced the numbers present with a new number sampled from a uniform distribution of [0,10000].

For example

Sentence: Mariela was in the hospital, and she got 403 get well cards from around the country. When she got home she got 287 more cards from friends and family. Mariela got _____ get well cards.
Label: 403 + 287
Post transformation: Mariela was in the hospital, and she got 553 get well cards from around the country. When she got home she got 665 more cards from friends and family. Mariela got _____ get well cards.
Label: 553 + 665

Models

We used GPT3 (code-cushman-001, text-ada-001, text-babbage-001, text-curie-001, text-davinci-001, text-davinci-002, text-davinci-003, code-davinci-002), GPT1(base), GPT2(small), and T5(base) for our experiments.

3. Experiments

We performed three experiments. E1: GPT3 on 4 NumGLUE tasks using Formal Chain of Reasoning, E2: Prompt Tuning, and E3: Smaller models on Task 4 with new condensed prompt.

E1 focused on exploring the "Formal Chain of Reasoning" prompting approaches for NumGLUE tasks. Our experiments are geared towards symbolic outputs of the model rather than a direct evaluation, which GPT3 is not good at. For E2, we focused on perfecting the prompting techniques by compacting them while improving the overall performance. E3 was focused on using the refined prompt to evaluate the smaller models. Here we evaluated smaller models with fine tuning and tuning using original train data as synthetically created train data.

E1: GPT3 on 4 NumGLUE tasks

In this experiment, we test our "Formal Chain of Reasoning" prompting technique on 4 tasks of NumGLUE, namely TASK 1, TASK 3, TASK 4, and TASK 8. Our approach is inspired by the chain of thought prompting and symbolic reasoning. Given a problem, we first extract information from the problem description. Next, we describe the relation by equations, and once we have all the equations available, we strip symbols of their meaning and derive answers by symbol manipulation. We apply a similar mechanism for solving an arithmetic problem. The first step would be to use a language model to identify the numbers and associated entities. Then, the model would derive equations from the relations between those numbers and entities. Once we have all the equations available, we can feed them to an arithmetic solver program to perform the calculation and provide the required answer.

Below is an example showing the same:

Input to GPT3 (Along with Prompt):
A shopkeeper has 7 decks of playing cards. How many red color cards does he have in total?
Output from GPT3:
r = 26, result = 7 * r
The output would be passed to a solver, which evaluates the result and returns 182.

E2: Prompt Tuning

Given the success of GPT3 on Task 4. We wanted to explore and perfect the prompting technique. We experimented with making the prompt more verbose and shortened to keep the performance but condense the overall text. We evaluated a range of GPT3 models to see how the performance of each prompt scaled with model size.

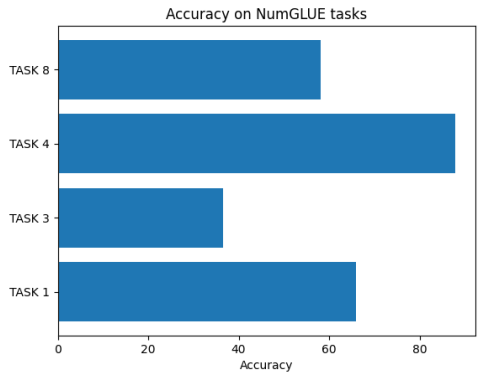
E3: Smaller models on Task 4 with new condensed prompt

Equipped with a refined prompt, we evaluated smaller models on the prompt. On the refined prompt, we evaluated smaller models: GPT1, GPT2, and T5. We evaluated the model by finetuning it on Numglue and synthetically generated data using the method described in the Methods section. We hypothesized that the untrained models would continue the scaling trend seen with the GPT3 models. For the fine-tuned models, we hypothesized that we could boost the performance on the Task.

4. Results and Analysis

E1: GPT3 on 4 NumGLUE tasks using Formal Chain of Reasoning

For 3 out of the 4 tasks that we evaluated on our prompting approach, we are getting significantly better performance compared to models using GPT3 are end-to-end models. Further, the technique does not rely on GPT3 to solve the equations, so our approach scales very well with large numbers. Another advantage of our approach is that it generates explanations and equations that help analyze why the model fails for specific input. For the tasks, we observed that the model needed help understanding concepts like time in a 12 or 24-hour format, problems involving linear equations, and the relation of friction and heat. In the end, the prompt was derived from a handful of inputs from training data randomly picked and kept if GPT3 could not generate the correct answer for them. Through this approach, we were able to get maximum accuracy while minimizing the size of the prompt.



E2: Prompt Tuning

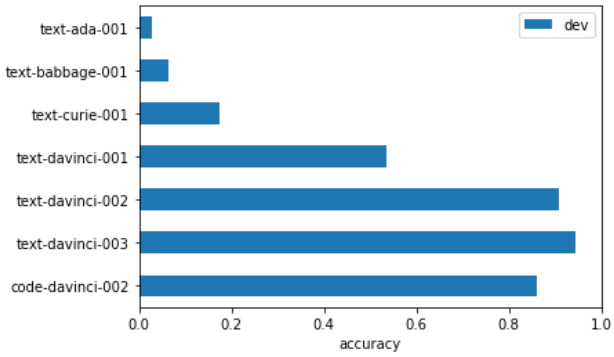
We restricted our attention to Task 4. We wanted to test whether an explicit step-by-step chain of reasoning drives the prompt's performance. For this experiment, we 1) eliminated all formal chain or reasoning steps except for the numerical equation and 2) evaluated it on smaller models.

To our surprise, removing the formal chain of reasoning steps boosted the performance across all models, especially in the smaller ones. For instance, babbage-001 increased by 10 percentage points from 0.064 to 0.164. The removal of the steps did not significantly affect the higher-performance models. The fruitful experimental results on the smaller models were strong evidence that training non-GPT3 models with this prompt might yield better results.

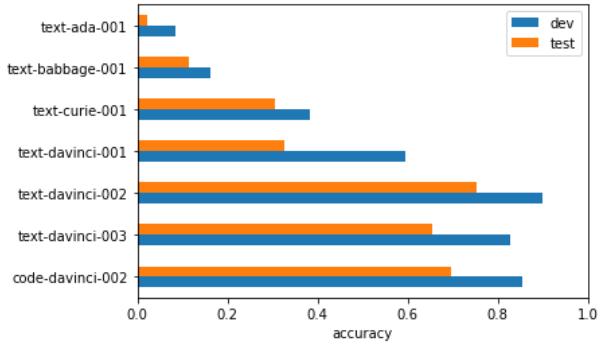
Another interesting result was that overall, code-davinci-002 performed worse than text-davinci-002. Since we engineered the prompt around text-davinci-002, we maximized its performance over other models with on-par parameters. We felt confident in evaluating the prompt on the test dataset. However, the observed performance was subpar compared to the dev set. The examples used might have been biased toward the train/dev set. We should have included more examples in the prompt to capture the test distribution better.

To summarize. For this task, a condensed prompt in the equation form yields better results. Larger models do not seem to benefit significantly from the verbose chain of reasoning prompt except text-davinci-003, which achieved a whopping performance of 0.94 accuracy on dev set, with an original prompt

Original Prompt



Shortened Prompt

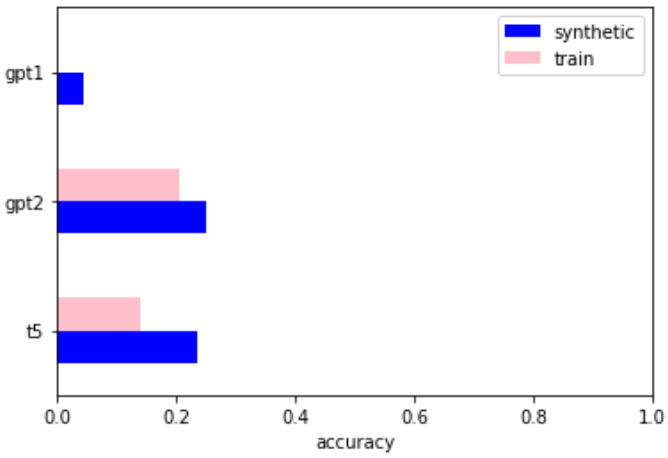


E3: Smaller models on Task 4 with new condensed prompt

Using the refined prompt along with the synthetically generated data from Numglue Task 4 training data, we tested the performance of GPT1, GP2, and T5.

As an initial experiment, we evaluated GPT1 and GPT2 using the original prompt without fine-tuning. However, neither model could produce any correct results on the test dataset. Then, we were able to tune it fine using train data as well as synthetic data. GPT1 did not produce any

results after the train data tune. However, we observed some correct answers when we tuned them on synthetic data. The accuracy was still less than 5%. GPT2, on the other hand, has some correct answers on train data and a small boost on synthetic data. We observed the same trend with T5. In the case of T5, however, the boost gap was larger, but the overall max accuracy did not exceed that of GPT2. Comparing the results to GPT2, it appears that the encoder does not contribute to the model's performance. In addition, we also want to point out that GPT1 and GPT2, in the case of synthetic data, performed better when trained on a small number of epochs. We observed worse performance when the models we exposed to more than five epochs of synthetic data.



5. Conclusions

We investigated a novel technique based on sampling and numerical output for a subset of NumGLUE tasks. We achieved SOTA on several tasks using the technique. We condensed and perfected the techniques and tested and evaluated them on smaller models in the GPT3 category and GPT2, GPT1, and T5. We found that prompting models scale linearly with the number of parameters; however, fine-tuning the smaller models significantly boosts smaller models.

6. Individual Contribution

We divided the prompt generation and task evaluation equally: Aakash Vora took on tasks 1 and 3, and Pavel Dolin took on tasks 4 and 8. Each of us designed a prompt specific to the task we were solving, inspired by our approach to generating equations. Finally, the report was a joint collaborative effort. After the intermediate review, Aakash generated synthetic data, and Pavel worked on code for plots and experiments. We both ran the experiments together and split the writing part equally. Overall we contributed to the project equally.

7. References

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of Thought Prompting Elicits Reasoning in Large Language Models. [2022-11-03]arXiv:2201.11903 [cs].

Swaroop Mishra, Arindam Mitra, Neeraj Varshney, Bhavdeep Sachdeva, Peter Clark, Chitta Baral, and Ashwin Kalyan. 2022. NumGLUE: A Suite of Fundamental yet Challenging Mathematical Reasoning Tasks. [2022-09-06]arXiv:2204.05660 [cs].

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of Thought Prompting Elicits Reasoning in Large Language Models. [2022-11-03]arXiv:2201.11903 [cs].

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2022. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. [2022-11-03]arXiv:2205.10625 [cs].

8. Appendix

Literature Review

GPT3 Prompting Techniques

Nowadays, Prompting language models has become so common that people have coined the term “Prompt Engineering” to represent the method of finding a prompt to solve a particular problem. Since our approach is about finding a prompting technique that can guide GPT3 to solve logical and numerical problems, we would like to highlight two recent prompting techniques used for the same tasks.

Chain-of-Thought (Wei et al., 2022)

This prompting technique hinges on adding intermediate explanatory natural language sentences between the input and output. The prompt consists of triplets <input, chain of thought, output>. Authors demonstrated that this prompting technique boosts LM models on tasks involving arithmetic, commonsense, and symbolic reasoning. Although the method is simple and powerful, authors report that sensitivity to examples has a vital role in performance. The method has pitfalls when it comes to the consistency of the performance. For instance, a permutation of the examples can cause the performance to range from a coin flip to state-of-the-art.

Least to most (Zhou et al., 2022)

Least-to-most is a recent prompting technique that has shown promising results for solving reasoning problems using GPT3 and surpassed the Chain-of-Thought prompting technique. This

technique divides problem-solving into two steps: 1) Problem reduction and 2) subproblem solving. During the first step, GPT3 is used to identify all the related subproblems that should be solved to solve the actual problem. In the next step, all the identified subproblems are solved using GPT3 and combined to find the final solution. Though this technique has provided promising results, some limitations seem to exist. One of the significant limitations is GPT3's poor performance in large numbers evaluation. Even though we can figure out the subproblems using this technique, the problem-solving step might return a wrong answer. The second limitation is that it generates output in natural language, so the output can not be passed directly to the downstream systems.

Prevalent approaches to solving arithmetic problems

With the advent of transformers, many problems in NLP have been reduced to end-to-end deep learning models. Earlier, the models developed for solving any NLP problems included a lot of modules working together, each solving a subproblem. However, most of the common problems in NLP, like POS Tagging, Name-entity recognition, etc., have been solved end-to-end by transformer-based large language models. Moreover, the transformers also solved the language translation problem with a single, large deep learning model, which makes it a perfect example to highlight the capabilities of the deep learning system.

However, there are a few problems where end-to-end models still need to achieve satisfying performance. One such class of problems is related to logical and numerical reasoning. Logical and numerical reasoning is a major ingredient for creating systems with natural language understanding. Deep learning approaches have mastered the problems requiring pattern recognition, but when we use them to solve problems requiring basic reasoning skills, they tend to fail miserably. Even though large language models like GPT3 have shown promising results for a task requiring some reasoning, it is highly unreliable and susceptible to attacks like replacing numbers with unreasonably large values or slight rephasing of the sentence.

Finding new models that are as expressive as these language models and rigorous enough to reason correctly in a sound and robust manner has become necessary. This observation has become the foundation for our approach, which we will discuss in the next section.

Taskwise Analysis of GPT3 on 4 NumGLUE tasks using Formal Chain of Reasoning

TASK	Accuracy (%)	Observations

TASK 1 (Prompt) (Evaluation)	66	After analyzing the results, we observed that GPT3 is performing exceptionally well at generating equations but has a weak understanding of some concepts. A few such concepts are the conversion of metric for comparison, time in 12 hrs and 24 hrs format, and equations with two variables. The proposed improvement on this task would be to change the prompt so we can explain such concepts to GPT3 so that it can take care of them while generating equations.
TASK 3 (Prompt) (Evaluation)	36.5	We used multi-step derivation for this problem where GPT3 identifies the fact and relation in the first iteration. Then GPT3 is again prompted with the output to generate equations based on predicate calculus. However, as it can be observed, the task's accuracy is very low for the following reasons: 1. GPT3 does not understand general concepts like heat, friction, or conversion between metrics. 2. GPT3-generated output runs into syntax issues; hence we believe that the codex model might give better results.
TASK 4 (Prompt) (Evaluation)	87.9	We observed unexpected success with this task, achieving a whopping 87.9% accuracy. Examining closer, the question shares the same structure and does not require multi-step reasoning. GPT3, with a given prompt, successfully identifies proper variables and "strings" them together with an appropriate set of arithmetic operators. Moreover, PT3 is good at combining those steps.
TASK 8 (Prompt) (Evaluation)	58.1	The results are mixed. The model fails to write the system of equations correctly. Sometimes the variables are not properly substituted or the equations are combined correctly. In some cases the system of equations is nor set up correctly. Further prompt engineering is required.

Supplementary Figures

Task	Question Setting	Size	Example
TASK 1	Commonsense + Arithmetic	404	Question: A man can lift one box in each of his hands. How many boxes can a group of 5 people hold in total? Answer: 10
TASK 2	Domain specific + Arithmetic	1620	Question: How many units of H_2 are required to react with 2 units of C_2H_4 to form 2 units of C_2H_6? Answer: 2
TASK 3	Commonsense + Quantitative	807	Question: A person wants to get shopping done quickly. They know that they can get through the check-out at big store in 5 minutes whereas it can take 20 minutes at small store. The store they go to finish quickly is? (A) big store (B) small store? Answer: big store
TASK 4	Fill-in-the-blanks	1100	Question: Joan found 70 seashells on the beach. She gave Sam some of her seashells. She has 27 seasshells left. She gave _____ seashells to Sam? Answer: 43
TASK 5	RC + Explicit Numerical Reasoning	54212	Passage: <>. Question: How many counties were added in 1887? Answer: 2
TASK 6	RC + Implicit Numerical Reasoning	32724	Passage: <>. Question: Which player kicked the shortest field goal? Answer: David Akers
TASK 7	Quantitative NLI	9702	Statement 1: James took a 3 - hour bike ride, Statement 2: James took a more than 1 - hour bike ride, Options: Entailment or contradiction or neutral?, Answer: Entailment
TASK 8	Arithmetic word problems	1266	Question: Joe had 50 toy cars. If he gives away 12 cars, how many cars will he have remaining?, Answer: 38

Table 1: Size and example of each task in the NumGLUE benchmark. RC: Reading Comprehension