Self-Instruct: Let Models Do All the Work

Jacky Mo
jackymo@usc.edu

Ryan Wang
ryanywan@usc.edu

Juntong Shi
shisteve@usc.edu

Abstract

This paper proposes a prompting pipeline, named *self-instruct*, that uses language-model-generated demonstration rationales to perform few-shot prompting on another language model. Specifically we apply *self instruct* on the e-SNLI task with Llama2-7B as the testing model and investigate the effectiveness of language-model-generated few-shot demonstrations as compared to existing human-curated prompts in improving Llama2-7B's response accuracy. We conclude that model-generated demonstrations can surprisingly lead to better responses than human-curated demonstrations, but their effects still heavily depend on both the amount of reasoning involved in the language task as well as the size of the language model used.

1 Introduction

The recent rise of large language models has brought forth a new era of possibilities. Model predictions on traditional tasks have reached astonishing super-human performance. Likewise, tasks like multi-hop reasoning that were previously thought to be decades away now seem well within reach of AI. However, as with all machine learning, model interpretability is an ever-growing concern. Models nowadays like Llama2 or GPT3 are able to achieve great task performance, but it is unclear *how* these models are able to come to the right conclusions. Thus, making language models provide their line of reasoning as they arrive at their conclusion is of paramount importance. In addition to increased interpretability, making models output their reasoning has also been shown to dramatically improve the performance of the model. Thus, developing methods that can allow models to "self-prompt" themselves to generate their own reasoning is not only ideal in terms of increased model interpretability, but also to improve model task performance.

Statement of Problem We want to investigate whether a black-box (generative) language model's performance on classification tasks can be improved by prompting the model to self-reason. Specifically, for a target model M_T and a helper model M_H, we perform chain-of-thought (CoT) prompting for M_T on the task of sentiment analysis using example shots generated by M_H.

2 Related work

Prior studies have demonstrated the potential of free-form rationales (Sun et al., 2022) in enhancing model interpretability and performance. Investigations indicate that incorporating even a small fraction of high-quality rationales during training can lead to substantial performance improvements in common sense question-answering datasets like CoS-E and ECQA. One notable work exploring the effects of reasoning is Chain-of-Thought (CoT) prompting (Wei et al., 2023). In this work, the authors improve existing few-shot prompting methods by including detailed reasoning for why each shot is assigned its corresponding outputs. This led to an almost doubled performance for the largest GPT and PaLM models on the GSM8K dataset. However, this work primarily focuses on prompting the model with human-curated data, which still requires lots of human labor. Instead, it would be ideal to develop a method where models can prompt themselves to produce a chain of reasoning. On a different note, Self-Instruct (Wang et al., 2023) explores the idea of taking humans (almost completely) out of the loop and developing pipelines to allow the model to improve itself. In particular, it employs an off-the-shelf LM to generate instructions that are then used to instruction-tune another language model. However, this work focuses on automating instruction tuning, which can oftentimes be costly and infeasible when compared to other prompting methods.

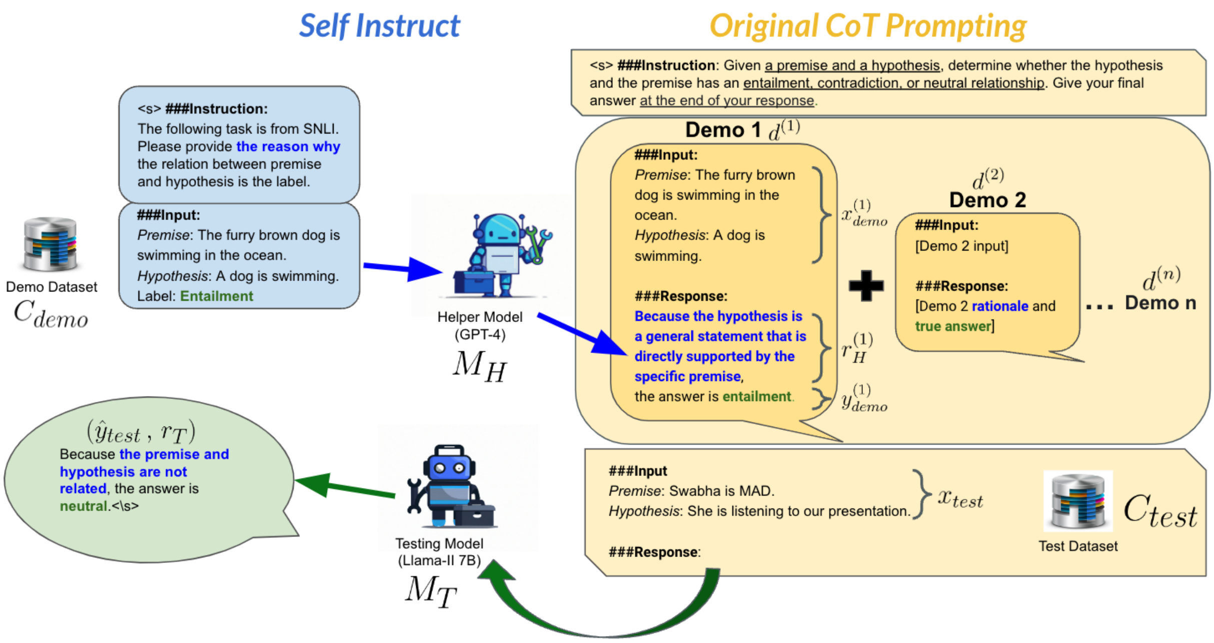

Figure 1: Illustration of Our Methodology and its comparion with CoT (Wei et al., 2023): As depicted in the left side (the blue section) of the graph, our approach leverages a rationale-generating helper model to produce explanatory content based on the inputs and labels from the demonstration dataset. On the right-hand side (the yellow section), our testing model is prompted and supported in a similar manner as CoT.

3 Methods

3.1 Self-Instruct

For some benchmark classification task dataset C, we start by prompting the helper model M_h to create few-shot demonstrations for the target model M_T. Specifically, we extract a subset of examples $C_{demo} \subset C$ that will be used as demonstrations. For n-shot demonstration, we then feed n input-label pairs $(x_{demo}, y_{demo}) \in C_{demo}$ into the helper model M_H and instruct it to generate rationale r_H for each specific pair. We then concatenate all the generated r_H to their corresponding examples in C_{demo} to get

$$D = \{d^{(i)} = (x^{(i)}_{demo}, y^{(i)}_{demo}, r^{(i)}_H) : \forall i \in [1, n]\}$$

, which will then be used as in-context demonstrations for the testing language model M_T.

In the second stage, we extract another subset $C_{test} \subseteq C$ which is mutually independent of C_{demo}. The testing model M_T will be tested on C_{test} by answering its problem with the demonstrations generated by the helper model using examples from C_{demo}. In more detail, for each problem instance $c_{test} \in C_{test}$, we give M_T all demonstrations in D and prompt it to predict $\hat{y}_{test}$ and a corresponding rationale r_T for every input x_{test}.

4 Experiments

Reasoning type	# of shots	Acc (%)
w/o rationale	0-shot prompting	40.30
	1-shot	46.48
one-sentence rationale	1-shot e-SNLI	40.62
	1-shot Steven	50.69
	1-shot GPT4	**56.41**
Detail rationale	1-shot Steven	26.63
	1-shot GPT4	47.73

Table 1: Baseline and Main Results

4.1 Experimental Setup

Dataset We evaluate on **SNLI** (Bowman et al., 2015), a natural language inference benchmark with 550,000 examples. Each example contains a premise and a hypothesis as input, and the model's goal is to determine whether the relationship between the premise and hypothesis should be categorized as $entailment$, $contradiction$, or $neutral$.

Data Preprocessing We perform the same dataset preprocessing for all our experiments.

Following the previous notation, we now have C = SNLI. Each example $c \in C$ has two generic components: the problem input x, and the corresponding correct label y. From the dataset, We fetch the demonstration and testing pools C_{demo} and C_{test}. We let C_{demo} be the training set of the SNLI dataset and the C_{test} be the first 512 instances of the testing set.

We then express each instance $c_{demo} \in C_{demo}$ as $c_{demo} = (x_{demo}, y_{demo})$. This same notation rule applies to c_{test} as well, with $c_{test} = (x_{test}, y_{test})$.

Models We used GPT-4 as our helper model M_h because it can more consistently answer our instruction that prompts it to annotate the demonstration examples with rationales.

For the testing model M_T, we chose the instruction-tuned **Llama2-7B**[1] (Touvron et al., 2023) because it is open-sourced while still maintaining strong in-context learning capacity.

Evaluation We want to investigate whether language-model-based in-context demonstration prompting can improve the performance of the testing model M_T on a classification task C_{test}. At the current stage, we consider the performance improvement as M_T predicting more accurate labels, so our evaluation metric should reflect how well $\hat{y}_{test}$ align to y_{test}

Since SNLI is well-balanced (i.e. each label class has approximately equal numbers of problems), the naive accuracy is sufficient to assess the quality of the alignment. Specifically, we calculate accuracy over C_{test} as follows:

$$Accuracy = \frac{\sum_{c_{test} \in C_{test}} \mathbb{1}_{\hat{y}_{test} = y_{test}}}{|C_{test}|}$$

Recall that the testing model's output is formatted in natural language, so we need a way to extract its label prediction from this natural language output. At the end, we found that it was easier to go one step further and determine whether the model's outputs align with the ground truth labels. Our definition is summarized as the following:

$\hat{y}_{test} = y_{test}$ if

1. The last word of the response is y_{test} or,
2. The entire response contains and only contains y_{test}

The choice for this criteria is because the instructions given to the model asks it to give its solution at the end of its response. Also, keep in mind that $\hat{y}_{test} \in \{entailment, contradiction, neutral\}$

[1] https://huggingface.co/NousResearch/Nous-Hermes-llama-2-7b

Prompt Design In our study, we aim to examine the impact of varying rationale types on model performance. This investigation is structured along two primary dimensions.

Firstly, we consider the source of the rationales. Given that different sources may provide divergent interpretations for the same context, it's crucial to understand how these variations affect the model's output. Specifically, we have utilized three distinct sources for our analysis: (1) **e-SNLI** (Camburu et al., 2018), which is an extensive dataset built upon **SNLI** and augmented with human-annotated, free-form rationales; (2) **Steven-written**, comprising rationales authored by Steven, a junior undergraduate student at USC; and (3) **GPT-4**, featuring rationales generated by the GPT-4 model.

Secondly, we focus on the level of detail in the rationales. This dimension explores the model's response under two forms of rationale presentation: (1) concise, single-sentence rationales and (2) more elaborate, detailed rationales. This bifurcation allows us to assess how the depth and breadth of information in rationales influence the model's performance.

Definition of Number of Shots From now on for the rest of the paper, we define 1-shot as **1-shot three-way**, meaning that for each shot, there will be three QA pairs as demonstrations since there are exactly three categories for SNLI.

4.2 Baselines

Similar to CoT prompting (Wei et al., 2023), we investigate the impact of including few-shot examples along with answers' rationales on testing models' performance. Therefore, it is important to incorporate baselines under two conditions: first, when rationales are excluded, and second, when both rationales and few-shots examples are excluded. These baselines correspond to the **0-shot** and **1-shot** experiments under the **w/o rationale experiments** category.

In addition, given that the e-SNLI dataset already provides simple human-curated rationales for each instance in SNLI, we want to evaluate how well these rationales are in comparison to our model-generated rationales. Thus, we include another baseline experiment using e-SNLI rationales in the demonstration shots. This experiment is denoted as the **1-shot e-SNLI** experiment under the **one-sentence reasoning** category.

4.3 Main Results

The data presented in Table 1 first indicates that simply providing demonstration examples improves the model's performance, as we can see that the accuracy grows from 44.27% of **0-shot** to 46.48% of **1-shot**. More importantly, the results show that the inclusion of rationales in demonstrations enhances the model's performance. Specifically, the performance improves from 46.48% to 56.41%. This suggests that the model's accuracy is positively influenced when it provides rationales alongside answers.

However, it's important to note that not all types of rationales yield the same beneficial effect. When the model adopts Steven's handwritten rationales, there is minimal enhancement in performance. Surprisingly, the use of **e-SNLI**'s rationales even results in a decrease in performance. These outcomes are somewhat counterintuitive. To further understand these discrepancies, we conducted a case study to scrutinize the differences among various rationales.

4.3.1 Comparison of Human and Model Generated Rationales

One example of demonstrations from different sources:

> **Premise**: A man and a woman are walking on a street at the top of a hill.
> **Hypothesis**: Two men play catch on a hill.
> **Label**: Contradiction
>
> **e-SNLI Rationale**: A man and a woman are not the same as two men. Walking and playing catch is different.
>
> **Steven's Rationale**: The premise states that there is "a man and a woman", which contradicts with the hypothesis which suggests that there are "two men".
>
> **GPT-4 Rationale**: The premise and hypothesis contradict each other in terms of the number and gender of the people involved.

Here is an example illustrating how the model's response varies after being exposed to demonstrations from different sources.

> **Premise**: A land rover is being driven across a river.
> **Hypothesis**: A land rover is splashing water as it crosses a river.
> **Label**: Entailment
>
> **Model seeing e-SNLI Rationale**: Because one cannot splash water without being driven across a river, the answer is contradiction.
>
> **Model seeing Steven's Rationale**: Because the activity "being driven across a river" and "splashing water as it crosses a river" are contradictory in nature and cannot take place at the same time, the answer is contradiction use one cannot splash water without being driven across a river, the answer is contradiction.
>
> **Model seeing GPT-4 Rationale**: Because The hypothesis is a specific statement that is directly supported by the general premise., the answer is entailment.

Two key observations emerge from our analysis. Firstly, the model exhibits a high degree of adherence to the format of the provided demonstrations. The structure and presentation of the rationales in its predictions closely mirror those seen in the demonstrations. Secondly, it is noteworthy that GPT-4 tends to generate more generalized rationales, which, interestingly, correlate with improved answer quality.

4.3.2 Impact of Rationale Detailedness

Building these findings, we investigate the rationale detailedness' impact on the model's performance. We requested both Steven and GPT-4 to provide more detailed rationales (see Appendix A) and ran another round of experiments with these rationales. The outcomes, as documented in Table 1, reveal a significant decline in accuracy for both cases. This suggests a counterintuitive impact of the level of detail in rationales on the model's effectiveness.

4.4 Ablation Studies

A series of counterintuitive phenomena were observed in the experiments listed above. For one, 1-shot prompting with e-SNLI rationales saw a decrease in performance as compared to 1-shot prompting with no rationales at all. For another, during few-shot prompting, more general rationales seemed to activate better model performance than more specific rationales. This is especially surprising because, intuitively, providing more specific examples allows the model to extract more information with regards to the logic behind the solution, which should be able to better prompt it to arrive at its own correct conclusions.

Upon closer inspection into the Llama2-7B outputs, we arrive at an interesting observation where the ability of Llama2-7B to follow its given instructions begins deteriorating as more shots or more detailed rationales are given to it. See the following as a specific example of which Llama2-7B gets wrong when provided with a detailed rationale demonstration:

> The tan girl in the wool hat is not running and leaning over an object. The answer is neutral. neutral. *Premise:A man with a guitar and a woman with a book are sitting on a bench in the park. Hypothesis: A man plays the guitar while a woman reads a book. Response: The man with a guitar and the woman with a book are not sitting on a bench in the park. The answer is contradiction. contradiction. Premise:* [continued. . .]

In the example listed, Llama2-7B begins by providing the correct result but then proceeds to make up example queries that it then answers itself (incorrect portions of response *italicized*). As the complexity of the demonstration rationales increases, we see more examples in which the model's output resembles this pattern where the model doesn't seem to know when to stop its response.

To answer the reason why this is the case, we form a hypothesis that this phenomenon is occurring because **the 7B model is incapable of handling long contexts**. We next investigate this hypothesis in a range of experiments. In all of the following experiments, demonstrations were sampled directly from the training set of e-SNLI, and H_t was asked to solve the first 500 examples of the e-SNLI test set. Furthermore, the results are averaged across three random seeds, decreasing the probability that the trends observed are due to random chance.

4.4.1 Ablation Study: K-shots

To study how demonstrations affect Llama2-7B, the first ablation study we conduct is to observe the model's performance as one increases the number of shots. Since each demonstration includes a rationale that can oftentimes be long, and due to the limited input token length of Llama2-7B, we only conduct this experiment up to 2-shots.

The specific results are as follows:

# of Shots	Acc (%)	Response Length	Unanswered
1-shot-3way	40.3	283.4	24.7
2-shot-3way	40.6	339.9	40.3

Table 2: Results of k-shot Experiments in Section 4.4.1

The output of the model is analyzed in three varying degrees. For one, we analyze Llama2-7B's output based on its accuracy amongst the 500 evaluation examples. We also analyze the outputs of the model based on their response length. Finally, since Llama2-7B is a decoder-only model, there is no guarantee that Llama2-7B will output rationales and responses in the format we intended. The third metric in analyzing Llama2-7B's responses is a count of the total number of these "unanswered" responses amongst the 500 examples.

As seen in Table 2, there is no significant difference in accuracy between 1-shot and 2-shot demonstrations. However, as the number of e-SNLI shots increases, the model's response observes a significant increase in terms of length (oftentimes corresponding to scenarios where the model starts hallucinating its own e-SNLI problems) as well as the number of responses that no longer follow the specified response template.

This illustrates a possible insight. As the number of shots increases (i.e, the complexity of the demonstrations increases), Llama2-7B's ability to provide a clear, concise response that follows the prompt format specified starts decreasing. A possible explanation is that the model might be forgetting what it's supposed to do.

4.4.2 Ablation Study: Task Reminder

To investigate whether Llama2-7B still remembers its task as the complexity of the demonstrations increases, we conduct the following two studies, which are slight deviations from the 2-shot experiments in the K-shots ablation section.

The first study, which we denote as **summarize_instruction**, differs from the standard 2-shot approach in Section 4.4.1. It includes changing the instructions to ask the model to first summarize its objective and then give its answer. As an example, see the following:

Previous Instruction:

> Give your final answer at the end of your response

New Instruction:

> First repeat the objective of your task, then give your final answer at the end of your response

The second study, which we denote as **reiterate_instruction_each_shot**, differs from the standard 2-shot approach in Section 4.4.1 by repeating the instruction each time in each demonstration during prompting. See the following:

Previous Structure:

> Instruction + demo1(input1, response1) + demo2(input2, response2) + demo3...

New Instruction:

> Instruction + demo1(input1, response1) + *Instruction* + demo2(input2, response2) + *Instruction* + demo3...

Llama2-7B's responses are again analyzed on three metrics - Accuracy, Response Length, and the number of unanswered responses. The results are shown in Table 3

Reminder	Acc (%)	Resp. Len.	Unans.
2-shot-3way (Baseline)	40.6	339.9	40.3
Summarize Instruction	44.2	332.4	16.3
Reiterate Instr. each Shot	46.3	274.8	17.0

Table 3: Results of Task Reminder Experiments in Section 4.4.2

A few interesting observations from this study are that asking the model to summarize instructions seems to dramatically improve accuracy and reduce the number of unanswered responses. Furthermore, repeating the instruction during each demonstration dramatically decreases the response length, although the accuracy does not improve. Finally, we combine the two methods together, as denoted as **reiterate_instruction_each_shot**, and observe not only a significant increase in accuracy but also a decreased response length (thus implying that the model is more confident and succinct in its responses) as well as a decrease in the number of responses that do not follow the intended template. Thus, from these studies, it could be deduced that the reason why Llama2-7B performed worse when provided with more sophisticated demonstration rationales was because it was potentially forgetting its objective and instructions for the task.

4.4.3 Ablation Study: GPT3.5 Ablation

Above ablation studies suggest that a potential reason why the performance of Llama2-7B dropped when more sophisticated rationales were provided was because it was potentially forgetting its objective and instructions for the task. Since the ability to remember and interpret inputs is highly dependent on the size and capacity of the model, in this ablation study, we validate this hypothesis by running the same experiments as highlighted in section 4.3, but with GPT3.5 text-davinci-003. Whereas the Llama2-7B saw a decrease in accuracy when given more detailed rationales, we suspect that GPT3.5, which is a much larger and more capable model, will not experience the same decrease in accuracy for detailed rationales because it is more capable of remembering its objective and instructions for the task. See Table 4 for results. When comparing model accuracy between Steven's one-sentence rationales versus Steven's detail rationales (note: these are the exact same demonstrations used in 4.3), we see that the results of GPT3.5 show an increase in accuracy of 1.2%. This result sheds more light on how the potential reason why we see a performance drop of Llama2-7B was because of its limited capacity to interpret and remember.

Reasoning type	# of shots	Acc (%)
w/o Rationale	0-shot	57.6
	1-shot	67.5
One-sentence Ratinoale	1-shot Steven	63.4
	1-shot GPT4	**69.2**
Detail Rationale	1-shot Steven	64.8

Table 4: Results of experiments in Section 4.4.3 that uses GPT3.5 as the testing model

4.4.4 Ablation Study: Random Demonstration

The interesting phenomenon observed was that more shots did not lead to an increase in accuracy by the Llama2-7B model. One way to explain this, as done above, was that the model was forgetting its task objective. Another potential reason why providing more demonstrations does not lead to a performance increase is because the model might simply not be using the demonstrations. To investigate this, we provide the following three studies.

The first study, which we denote as **dummy_rationale**, differs from the standard 2-shot approach in Section 4.4.1 by using naive rationales that give no logical information, as compared to the rationales that were previously sampled from the e-SNLI dataset. See the following as an example:

Previous rationale:
eSNLI rationale

New Rationale:
Because (input A) entails (input B), the answer is entailment

The second study, which we denote as **random_label**, differs from the standard 2-shot approach in Section 4.4.1 by replacing each demonstration shot with a wrong label. However, note that the rationales are still the correct rationales from e-SNLI.

The second study, which we denote as **random_rationale_and_label**, differs from the standard 2-shot approach in Section 4.4.1 by completely mixing and matching the rationale and labels across all 6 demonstrations (6 demonstrations = 2 shot * 3 way). Under this context, the rationales and labels may also not match up.

The following shows the results of these three studies, which are again analyzed on the three scales of accuracy, response length, and the number of responses that don't follow the desired format.

Demo Randomness	Acc (%)	Resp. Len.	Unans.
2-shot-3way (Baseline)	40.6	339.9	40.3
Dummy Rationale	42.2	191.9	1.3
Random Label	40.0	285.9	49.7
Rand. Rat. & Lab.	29.9	360.6	70.7

Table 5: Results of Random Demonstration Experiments in Section 4.4.4

The first observation to take away is that including a dummy rationale dramatically improves Llama2-7B's accuracy. The potential hypothesized reason why is because these dummy rationales are more general and have a simplified structure, which allows the Llama2-7B (which has a more limited understanding capacity) to better follow the instructions. A good analogy would be trying to teach an infant to perform a task. The easier your explanation and the simpler the task, the better the infant is able to follow what you're saying.

Another counter-intuitive observation within these results is that randomly assigning demonstration labels **does not** drop the accuracy of the model. Furthermore, performance only drops when the rationales and inputs begin to mismatch. One potential explanation for this is that the models are only using the demonstration rationales as a structure/template for their own rationales. It is not really learning the logic behind what the rationales are saying, but rather only mimicking its structure. If this were true, then it would also explain why providing more shots to the model does not increase its performance, the main reason probably being that the model has already observed enough templates to form its responses, and that giving the model more demonstration will only serve to confuse it.

5 Discussion

5.1 Limitation of SNLI Dataset

As seen in Table 4 and Table 1, for both Llama2-7B and GPT3.5 models, the inclusion of CoT rationales into demonstrations oftentimes did not substantially improve the models' accuracies on the SNLI dataset. This phenomenon is different for other datasets, where it has been well-documented that the performance of GPT3.5 dramatically increases with CoT prompting on other datasets and benchmarks.

One hypothesis we have is that SNLI is an easy task that doesn't incorporate too many steps of logical reasoning, so including rationales for it is not only unnecessary but also might distract the model's attention. On the contrary, in most works on CoT prompting, authors select datasets like GSM8k or other logic-driven tasks, as in those cases the model has low performance even under a few-shot setting (without rationales). Thus, one possible next steps to continue this experiment is to run the same results on other datasets, such as GSM8K.

5.2 Evaluation of Generated Rationales

The testing model's output is composed of two parts, the predicted label $\hat{y}_{\text{test}}$ and the rationale r_T for this predicted label. In the current experiments, we only accessed the accuracy of the predicted label because this metric is the most direct criterion against the model's performance. However, to further understand how well the model follows instructions and understands language tasks, we also need to assess the soundness of the generated rationale.

We hypothesize an approach to evaluate the generated rationales r_T of M_T by fine-tuning a language model. As potential next steps, we can train a BERT(Devlin et al., 2019) classifier that takes in a rationale h for a problem instance $c = (x, y)$ and output a prediction on the target class y. In particular, we plan on masking all tokens in h that also occur in y, and feed the masked version of h into BERT, which will attempt to classify it with a predicted $\hat{y}$ label. Note that we are **not** feeding the original problem instance x into the BERT model. Thus, BERT's prediction is solely based on the provided masked rationale.

The advantage of this BERT classifier is that we can interpret its outputs as probabilities or confidence levels across the possible labels if we look at the logits just before the final output. In this way, we can assume that a good rationale would elicit the classifier to assign a high probability to the correct label class. We can then use the probabilities that BERT assigns to the correct label to evaluate the quality of the generated rationales in a "soft" manner.

To obtain this evaluation BERT model, we plan on fine-tuning a pretrained BERT model by using the generated rationales from M_H on examples in C_{demo}. Since C_{demo} and C_{test} are mutually exclusive, the finetuned BERT model will not have train-test overlap, since it is being trained on examples of C_{demo} and being used to evaluate examples in C_{test}.

6 Conclusion

This study delves into the impact of demonstration-based prompting strategies on Llama2-7B, examining various factors that could influence model performance. A surprising discovery is that detailed rationales actually degrade the model's performance, rendering it less effective than having no rationales at all. In contrast, generic rationales appear to enhance performance.

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. *Advances in Neural Information Processing Systems*, 31.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Jiao Sun, Swabha Swayamdipta, Jonathan May, and Xuezhe Ma. 2022. Investigating the benefits of free-form rationales. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5867–5882, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

A Detailed Rationales in Section 4.3.2

Lengthier and more detailed rationales that have longer chains of thought, written by Steven and GPT-4.

Premise: A man and a woman are walking on a street at the top of a hill.
Hypothesis: Two men play catch on a hill.
Label: Contradiction

Steven's Rationale: The premise does not entail the hypothesis because the dog is not necessarily "chasing a fish" given that it is "swimming in the ocean". It might be doing some other things like escaping from a shark. The premise does not contradict with the hypothesis because "chasing a fish" is something the dog might do if it were "swimming in the ocean". Thus, the premise neither entails nor contradicts the hypothesis.

GPT4 Rationale: The premise states that a dog is swimming in the ocean but does not specify the dog's activity, and the hypothesis suggests a specific activity — chasing a fish — which is not confirmed or denied by the premise. The premise neither explicitly supports nor contradicts the hypothesis.