Efficient Translation of Natural Language to First-Order Logic Using Step-by-Step Distillation

Stanford CS224N Custom Project

Aliyan Ishfaq
Department of Computer Science
Stanford University
aliyan@stanford.edu

Shreyas Sharma
Department of Computer Science
Stanford University
shreyas3@stanford.edu

Abstract

The evaluation of large language models (LLMs) on natural language (NL) logical reasoning problems is crucial for downstream tasks like mathematical problem-solving, automated decision-making, and planning. A recent framework, LogicLM, outperforms GPT-4 on reasoning benchmarks by initially translating the NL reasoning problem into some form of formal logic statements, and subsequently solves them using deterministic symbolic logic solvers. However, challenges persist, notably in the accuracy of the initial translations to formal logic. Our project investigates the efficacy of Google's Step-by-Step Distillation (SBSD) method for fine-tuning a smaller model, CodeLLaMa-7B on the task of translating NL statements into their equivalent first-order logic (FOL) statements. As part of the SBSD approach, we prompt GPT-4o to generate rationales for NL-FOL pairs. To increase diversity in the logical structure of FOL examples for few-shotting GPT-4o's rationale generation, we implement an original clustering-based method. We find that our approach does not outperform the existing benchmarks for a 7B model on NL-FOL translation set by LogicLLaMa which uses standard fine-tuning. We contribute granular analysis of the types of errors our model makes and argue that this task is limited by the subjectivity of its evaluation methods. Finally, we perform ablation studies to understand the impact of number of epochs and dataset size and accuracy. Through this, we find promising, albeit currently inconclusive, results indicating potential for future SBSD-related experiments to achieve comparable performance with 30% of the training data.

1 Key Information to include

- Mentor: Ryan Li
- Team Contributions: Aliyan pre-processed the datasets, set-up and conducted multi-task fine-tuning, ran and analyzed the evaluations, and reported the results. Shreyas implemented the clustering-based few-shot example sampling method, wrote prompts for rationale-generation and fine-tuning, conducted the error analysis and completed the write-up.

2 Introduction

The ability to reason logically is a critical metric for evaluating large language models (LLMs). Logical reasoning is crucial for a substantial number of important tasks including, but not limited to, mathematical problem-solving, automated decision making, and planning. Currently, state-of-the-art performance on LLM reasoning is limited by the inability of LLMs to draw accurate faithful conclusions from provided premises. A recent approach, LogicLM by Pan et al. (2023), mitigates this issue by first using LLMs to translate reasoning problems in natural language to formal logic statements and then using deterministic symbolic logic solvers to infer accurate conclusions. Despite these advances, LogicLM falls short of achieving perfect scores on reasoning benchmarks, primarily due to inaccuracies in the initial translation by the LLM. Additionally, their experiments are limited to large, closed-source LLMs. LogicLLaMa Yang et al. (2023) addresses the translation of natural language statements to first-order logic statements using standard fine-tuning, achieving performance comparable to GPT-4. Improvement in NL-FOL translation is likely to be directly proportional to improvement of LogicLM's performance on reasoning questions related to FOL. Consequently, we see that accurate NL-FOL translation can stand to improve LLM-aided state-of-the-art reasoning capabilities. In addition to accuracy, an important desiderata is accessible NL-FOL translation. Thus, we focus on investigating NL-FOL translation by CodeLLaMa-7B, a small, inexpensive, open-source model. Our research aims to surpass the benchmark set by LogicLLaMa by investigating an alternative method. We adapt Google's step-by-step distillation (SBSD) method by Hsieh et al. (2023), which involves generating rationales for NL-FOL pairs and fine-tuning CodeLLaMA-7B on the multi-task problem of simultaneously predicting the label (the FOL statement) and the rationale. Although our approach achieved performance close to LogicLLaMa, it did not surpass it. Nonetheless, our contributions include: (1) assessing the efficacy of SBSD for the NL-FOL task compared to standard fine-tuning (2) Clustering-based Diverse Search of FOL Examples (CSFE), our original method to increase diversity in the logical structure of FOL examples for few-shotting LLMs for FOL-related tasks (3) conducting a granular analysis of specific errors made by our fine-tuned LLM on the NL-FOL task and consequently arguing the need for more nuanced NL-FOL evaluation metrics, and (4) performing ablation studies to better understand the impact of epoch size, dataset size, and accuracy on performance, where we find preliminary results showing the potential of this approach to maintain strong performance with significantly less labeled data.

3 Related Work

3.1 LogicLM

Previous approaches to improving the reasoning abilities of LLMs include few-shot prompting, CoT prompting by Suzgun et al. (2023), least-to-most prompting by Zhou et al. (2023) and fine-tuning by Wang et al. (2023). Recently, the LogicLM framework by Pan et al. (2023) outperformed other approaches. It consists of the following steps.

- **Problem Formulation:** Depending on the type of reasoning problem, the LLM is instructed to convert the natural language reasoning problem into one of four archetypes of formal reasoning: deductive reasoning, first-order logic, constraint satisfaction, and analytic reasoning
- **Symbolic Reasoning:** The formal formulation produced by the LLM is deterministically, and consequently accurately, solved using the corresponding suitable symbolic logic solver. For first-order logic, this is employing a first-order logic prover.
- **Result Interpretation:** The output from the symbolic logic solver is processed by an LLM or a result interpreter, which then generates the final answer to the initial reasoning problem.

LogicLM with GPT-4 outperforms GPT-4 with CoT prompting by 8.34% on the FOLIO dataset by Han et al. (2024). However, LogicLM only experiments with GPT-4 and GPT-3.5 as the LLM of choice for step (1). We are interested in investigating the performance of a smaller open-source model, specifically for the archetype of first-order-logic. Our motivation for this is the speed, affordability, and privacy afforded by smaller open-source language models.

3.2 LogicLLaMa

The approach closest to ours is LogicLLaMa by Yang et al. (2023). In the LogicLLaMa paper, LLaMa-7B was fine-tuned on 35K examples of NL-FOL translations for the task of NL-FOL translation.They use the MALLS dataset consisting of 34K synthetically generated examples of NL-FOL pairs and 1K pairs from the training set of LogicNLI by Tian et al. (2021). The authors Yang et al. (2023) also developed an evaluation metric, logical equivalence (LE) , to evaluate their performance and found that their fine-tuned version of LLaMa-7B was shy of GPT-4 5-shot by just 3.7% on LE.

3.3 Google's step-by-step distillation

Recently, Hsieh et al. (2023) at Google proposed a framework named Step-by-Step Distillation (SBSD) to improve performance on reasoning tasks, including natural language inference, commonsense question answering, and arithmetic word problems.
SBSD involves the following steps:

- **Rationale Generation:** Few-shot CoT prompting of a high-capacity LLM like GPT-4 to generate rationales that explain the reasoning process from question to label.
- **Multi-task Fine-tuning:** Fine-tuning a smaller model to simultaneously generate both rationales and labels.

Google's results show that SBSD outperforms standard fine-tuning using less training data for the aforementioned reasoning tasks by Hsieh et al. (2023). In our research, we adapt Google's SBSD method for a different reasoning task: NL-FOL translation. We compare our results against the performance of LogicLLaMa's standard fine-tuning approach.

4 Approach

Our modified version of Google's SBSD framework adapted to the NL-FOL translation task is as follows:

4.1 Rationale Generation using GPT-4o

First, we prompt GPT-4o to generate rationales given a NL-FOL pair. We use GPT-4o since it is strong at reasoning, fast, and inexpensive. Rationales are step-by-step explanations as to how to arrive at the given FOL translation based on the NL statement. We construct two different types of rationales, (1) Class-A and (2) Class-B.

4.1.1 Class-A Rationales

Our prompt for generating Class-A rationales is optimized using our original approach: clustering-based diverse selection of FOL examples for few-shot learning (CSFE). In few-shot prompting, variety in chosen examples is crucial for better generalization. Our method CSFE aims to diversify the logical structures of the examples chosen to few-shot GPT-4o. We create a dataset of masked/abstract FOL statements, replacing specific predicates and objects with generic labels (Ps and Os) while maintaining logical operators as is using regular expressions. For instance, $isYoung(daniel) \wedge isOld(danielsDad)$ is abstracted to $P(O) \wedge P(O)$. This approach ensures clustering focuses on the underlying logical structure rather than variable names. We implemented a tokenizer based on logical connectors, used a count vectorizer, and performed k-means clustering on the abstracted statements. From each cluster, we randomly sampled an NL-FOL pair. Qualitatively, we observed that our clustering algorithm successfully learned to form clusters of distinct logical structures. We observed that FOL statements in different clusters varied in terms of number of predicates, number of arguments, number of logical connectors, and types of logical connectors. However, there was a lack of variety with respect to statements that are existentially quantified, universally quantified, or neither, that is to say, there was no clear cluster where we could draw existentially-quantified statements or non-variable-based statements from. We will refer to this type of variety in FOL statements as Existential-Universal-Neither-diversity, or EUN-diversity for short. Since EUN-diversity would also be important for GPT-4o to generalize its performance, we decided to also draw samples such that the final samples had diversity with respect to cluster (and consequently, logical structure) as well as EUN-diversity. Finally, we include 3 diverse examples in our Class-A prompt.

4.1.2 Class-B Rationale

Due to the existence of multiple examples in the prompt, the Class-A prompt was long. Therefore, we augmented the Class-A rationales with Class-B rationales that only contained one randomly-selected example in the prompt.

Overall, we generated 2667 Class-A rationales and 5456 Class-B rationales, where the Class-A rationales were of higher quality due to an optimized prompt, but were also more expensive. Complete

Class-A and Class-B prompts can be found in Appendix A. The following is an example of a GPT-4o generated rationale:

4.2 Multi-task fine-tuning

After generating the rationales we simultaneously fine-tune CodeLLaMa-7B on both generating the label, i.e. FOL statement, as well as the rationale. We use CodeLLaMa since we hypothesize that CodeLLaMa's enhanced coding abilities will translate to enhanced logical problem-solving abilities. We use 7B parameters since it is much smaller than the models used by LogicLM (GPT-3.5 and GPT-4) and is also the same size as our base-line, LogicLLaMa. We conduct the fine-tuning on MALLS v0.1, consisting of 27K NL-FOL pairs, i.e. 27K question-label pairs as well as 8123 rationales. We ensure that the NL-FOL pairs that the 8123 rationales are based on are a strict subset of the 27K NL-FOL pairs. This ensures that the rationales cannot improve model performance by providing additional examples of NL-FOL pairs that the model can learn from. This is to keep our comparison to our baseline fairer.

Based on Google's success, we hypothesize that the rationales contain important and widely-applicable principles and reasoning steps of how to conduct NL-FOL translation that the model learns through the rationale-generation task. For instance, the model could learn that the natural language phrasing "If…, then…" requires the logical connector '$\rightarrow$' from our rationales. This is because in our rationale-generation prompts, we include the instruction to directly reference the NL statement when justifying a choice of logical operator. Like Google, we prepend the task prefixes [LABEL] and [RATIONALE] to the input data that we fine-tune on so as to allow the model to differentiate between the two tasks. We conduct fine-tuning on subsets of the MALLS dataset released by the LogicLLaMA authors.

4.3 Baselines and Evaluation

We compare our approach which uses fine-tuning with SBSD on CodeLLaMa-7B on MALLS v0.1 and 8123 rationales against our baseline of LogicLLaMa which uses standard fine-tuning (without SBSD) with LoRA on MALLS v0 and no rationales. MALLS v0.1 and MALLS v0 are similar synthetic NL-FOL datasets released by the LogicLLaMa authors. The difference is that the former has 27K data points which are auto-verified whereas the latter has 34K data-points without auto-verification. We use the former to increase accuracy of our training data, which is an important concern since the dataset is synthetically generated. We evaluate the models on FOLIO-parsed, a subset of the FOLIO dataset parsed by the LogicLLaMa authors. FOLIO-parsed contains 2195 human-expert-written NL-FOL pairs.

5 Experiments

5.1 Data

We use two datasets for our experiments: **MALLS v0.1** and **MALLS v0** by Yang et al. (2023).

MALLS v0.1: The dataset comprises 27,000 NL-FOL pairs that have been auto verified.

MALLS v0: The dataset includes a total of 34,000 NL-FOL pairs that are non verified.

Additionally, we generate 8,123 rationales from randomly sampled NL-FOL pairs within the MALLS v0.1 dataset. Of these, 2,667 rationales are Class-A rationales, that is to say they are generated using our CSFE approach, which aims to improve the diversity and representativeness of the rationale set.

Finally, we evaluate all models on the same dataset that LogicLLaMa is evaluated on, namely folio-parsed, a parsed version of the original FOLIO dataset by Han et al. (2024), parsed by the authors of the LogicLLaMa paper by Yang et al. (2023).

For our experiments, we construct three distinct training sets:

- MALLS v0.1 and Rationales - **D1**
- MALLS v0 and Rationales - **D2**
- 8,000 randomly-sampled NL-FOL pairs from MALLS v0.1 and Rationales - **D3**

Table 1: Sample Class A Rationale generated from NL-FOL pairs

Natural Language	First Order Logic	Rationales
If a person is a judge, they preside over court proceedings and make legal decisions.	$\forall x(\text{Person}(x) \land \text{Judge}(x) \rightarrow (\text{PresideOverCourtProceedings}(x) \land \text{MakeLegalDecisions}(x)))$	Identify Key Entities and Relationships: The key entities are "person" and "judge." The relationships are "preside over court proceedings" and "make legal decisions," which are conditions associated with being a judge. Use the variable x to represent a person. Logical Formulation Structure: $\forall x(\text{Person}(x) \land \text{Judge}(x) \rightarrow (\text{PresideOverCourtProceedings}(x) \land \text{MakeLegalDecisions}(x)))$ Explanation for Logical Formulation: The universal quantifier $\forall x$ ensures the statement applies to all persons, reflecting the generality introduced by the phrase "If a person." The predicate Person(x) and Judge(x) identify the individual as a person and specifically as a judge. The implication operator $\rightarrow$ is used to convey the conditional nature of the statement: if someone is a judge, then certain conditions follow. This aligns with the use of "if" in the natural language statement. The predicates PresideOverCourtProceedings(x) and MakeLegalDecisions(x) describe the activities associated with being a judge, directly corresponding to "preside over court proceedings" and "make legal decisions." The conjunction $\land$ connects these activities, reflecting the cumulative conditions laid out in the natural language statement. Therefore, the logical structure accurately translates the conditional requirements specified in the natural language into first order logic.

By comparing these training sets, we aim to evaluate the impact of dataset size and verification on model performance.

5.2 Evaluation method

The predicted FOL pairs are evaluated against the true FOL pairs using two metrics: BLEU and Logical Equivalence (LE).

BLEU: The BLEU score measures the ability of a language model to accurately extract relevant predicates and objects, and to make appropriate choices in naming them. The BLEU score is calculated using the following equation:

$$\text{BLEU} = \left(\prod_{n=1}^{N} p_n\right)^{1/N} \cdot \text{BP} \tag{1}$$

where p_n is the precision for n-grams, and BP is the brevity penalty.

Logical Equivalence (LE): The LE score evaluates the equivalence of two logical statements based on their truth tables, computing the overlap ratio. This metric compares the logical structure and meaning between the predicted and true FOL pairs.

We adopt LogicLlama's approach for evaluating both BLEU and LE scores. By comparing these metrics, we evaluate the performance of the models in terms of both linguistic closeness to True FOL and logical correctness in comparison to True FOL.

5.3 Experimental details

We fine-tuned the CodeLlama 2-7B model on the training sets with a learning rate of 3×10^{-5} and a batch size of 8. For generation, we used a maximum token limit of 100, a temperature setting of 0.7, top_k of 50, and top_p of 0.7.

All experiments were conducted on Together AI's cloud service using an A100 GPU.

We also experimented with the number of epochs to determine its effect on the model's performance, measured using the BLEU and Logical Equivalence (LE) metrics.

5.4 Results

The following table shows the results of fine-tuned CodeLlama 2-7B with different datasets and number of epochs on FOLIO Parsed, our test set. We compare our results to the Direct Translation fienetune results of Llama 2-7B from LogicLlama by Yang et al. (2023).

Dataset	Model Name	Epochs	Median BLEU	Mean BLEU	Median LE	Mean LE
D1	CodeLlama 2-7B	2	0.2320	0.3163	0.8125	0.7695
D2	CodeLlama 2-7B	1	0.2215	0.3134	0.7813	0.7629
D2	CodeLlama 2-7B	2	0.2215	0.3130	0.7813	0.7614
D2	CodeLlama 2-7B	3	0.2218	0.3157	0.8125	0.7636
D3	CodeLlama 2-7B	2	0.2215	0.3079	0.7813	0.7545
MALLS	Llama 2-7B (Direct Translation) [CITE]	-	-	0.3720	-	0.8180

Table 2: Results of Different Datasets and Models

6 Analysis

6.1 Ablation Result Analysis

From Table 2, we see that epoch size has minimal effect on accuracy. We see that using the 27K auto-verified dataset leads to a slight performance improvement over the 34K un-verified dataset.

Most interestingly, we see that in one of our experiments, despite reducing the number of FOL labels to 8K (D3), we maintain excellent performance, achieving 1% less LE than D1, with 30% of the labels. This is similar to Google's findings in their SBSD paper where they were able to leverage SBSD to achieve higher performance with significantly less training data. This result is especially promising since there are very few English NL-FOL datasets, and even fewer non-English datasets. Consequently, SBSD has the potential to augment equity in terms of language access for this task. However, while this result may be promising, it is unfortunately inconclusive at this point. When running our main experiment with 27K labels and 8123 rationales, we ensured that the 8123 rationales were based on FOL labels that were a strict subset of the 27K FOL labels to ensure that the rationales cannot provide any new FOL information. Unfortunately, we made the mistake of not enforcing this rule when conducting the aforementioned ablation experiments on the 8K labels subset and randomly selected the 8K labels instead. We attempted to re-run this ablation experiment with the rule enforcement, however our compute provider, TogetherAI, consistently failed to deploy our model. Nevertheless, the result of the experiment is still promising since in the unlikely worst case, if the sets of FOL statements for our labels and rationales were completely disjoint (which is unlikely), we would have 8000 + 8123 = 16123 distinct FOL statements, which is less than 27K, while maintaining comparable performance.

6.2 Error Analysis

We identify 2 categories of errors, E1, E2.

6.2.1 E1

Subjectivity of determining if an object is general enough to remain interpreted as an object, or if a predicate needs to exist to state that the object is an object of a specific type.

- Consider the following example. NL: 'All people who regularly drink coffee are dependent on caffeine.' Predicted FOL: $\forall x$ (Person(x) $\land$ RegularlyDrinksCoffee(x) $\rightarrow$ DependentOn-Caffeine(x)). True FOL: $\forall x$ (Drinks(x) $\rightarrow$ Dependent(x)).
- The interpretation of the predicted FOL is that for any object x, if x is a person and regularly drinks coffee, x is dependent on caffeine. The interpretation of the True FOL is that for any

object x if x drinks [coffee], then x is dependent [on caffeine]. We argue that our predicted FOL is in fact more accurate than the True FOL in both LE and BLEU. With respect to LE, we see that logically, it is not necessary that all objects are people. For example, the conclusion that a cat that drinks coffee is dependent on caffeine is inconclusive based on the NL statement since the NL statement only discusses people (humans). However, the True FOL, would (arguably) mistakenly mark the conclusion as true. Furthermore, despite our Predicted FOL being logically correct, we receive a lower LE score of instead of 1 due to the extra predicate compared to the True FOL. With respect to BLEU, we clearly see that our predicted FOL is in fact more expressive than the True FOL since we specify predicates RegularlyDrinksCoffee as opposed to simply Drinks. Thus, we believe we receive a lower BLEU score than deserved for this category of examples.

6.2.2 E2

Subjectivity of identifying universally quantified statements.

- Consider the following example: NL: Two is positive, Predicted FOL: $\forall x$ (Two(x) $\rightarrow$ Positive(x)), True FOL: $Positive(two)$.
- This type of error is admittedly more valid to penalize, since given just the context of the NL statement, the more straightforward FOL representation is the True FOL. However, the predicted FOL is interpreted as for all objects x, if x is two, then x is positive. This, while less straightforward, fundamentally has the same meaning as the True FOL, and should arguably not be penalized as much as it is.

Both E1 and E2 arguably arise from the inherent subjectivity of NL-FOL translation due to the existence of multiple correct FOLs for a given NL statement.

6.3 Baseline-Comparison Analysis

While we tried to make our comparisons fair, their still exist certain differences between LogicLLaMa and our approach that should be controlled for, such as (a) us using CodeLLaMa and LogicLLaMa using vanilla LLaMa, (b) LogicLLaMa training on an additional 1K examples from the LogicNLI dataset, (c) us using MALLS v0.1 and them using MALLS v0, and (d) different hyper-parameters for fine-tuning as well as their use of LoRA. Any combination of these differences could affect our conclusion of the effect of SBSD vis-a-vis standard fine-tuning. For more granular conclusions, further ablation studies are required.

7 Conclusion

In conclusion, while we were unable to beat the challenging baseline set by LogicLLaMa, we demonstrate the potential of the SBSD framework in achieving comparable performance with less data on the NL-FOL translation task. Our original clustering-based approach qualitatively demonstrates significant utility in achieving diversity for few-shot prompting for FOL-related tasks. Our error analysis contributes to the existing research by demonstrating the need for more nuanced evaluation metrics that take into account the subjective nature of NL-FOL translation. In the future, we intend on conducting further ablation studies robustly experimenting with the performance of SBSD on decreasing dataset sizes.

8 Ethics Statement

Our project raises two ethical concerns. Our first concern is the co-optation of the perception of FOL-based LLM-reasoning as infallible in order to obfuscate the subjectivity, subjective utility functions, and bias of oftentimes messy real-world decision-making scenarios. Entities with decision-making authority, such as governments, corporations, and academic institutions, might exploit the public's perception of the infallibility and apparent objectivity of First-Order Logic to legitimize their decisions. The issue with this perspective is that although LLM-based reasoning, as demonstrated in our approach, and frameworks like LogicLM can draw objectively correct conclusions, the objectivity of any reasoning/decision-making mechanism is only as objective as the axioms that govern said mechanism. In other words, the soundness of a conclusion does not necessitate the validity of the premises from which the conclusion was correctly derived from. So for instance, if a government starts with the assumption that economic growth is always good, then no matter how objective any reasoning mechanism is, it will always lead to conclusions in favor of economic growth. This is problematic because the aforementioned government can use these conclusions to justify economic growth at the cost of environmental protection, for instance. Ideally, such assumptions and axioms would be scrutinized by stakeholders. The ethical concern we anticipate is that these subjective assumptions may be incorrectly hand-waved away by powerful institutions who appease the people they impact with rhetoric like "How can you question this conclusion? It has been validated by this objective AI-logic framework!". The issue here is that many real-world situations requiring reasoning systems also require subjective value systems, subjective utility functions, and subjective objectives. This can be mitigated by transparently and accurately reporting the limitations of LLM reasoning and that any AI system is only as unbiased as the value system that it is explicitly aligned to allow it to be. Further mitigation can be achieved by openly disclosing the entire mechanism, including assumptions, models, and frameworks used for any automated reasoning or decision-making process.

Secondly, we recognize the substantial benefits possible from downstream applications of our translation task. Improved automated LLM reasoning and consequent decision-making can be used to optimize policy, automate reasoning work, and bring about public welfare. The second concern we raise is the potential and likely inequitable distribution of the aforementioned power. Our approach, and the approaches before us are English-focused. To mitigate potential biases, it is important to evaluate and optimize models for non-English languages as well, considering the significant variability in the task of NL-FOL translation across different languages. Initially, we sought to mitigate this bias by training models on the same task in other commonly-spoken languages. However, we encountered few ground-truth publicly available datasets of NL-FOL translation in other languages.Additionally, we recognized that unlike certain tasks, this issue cannot be circumvented by translating the task from a target language to English, running inference, and then translating back. This is due to the inherently subjective nature of predicate selection in FOL statements. For instance, a single natural language clause might be interpreted as either two simple predicates or one complex predicate. Such subjectivity is likely to be exacerbated by linguistic differences, such as the explicitness of verbs, the complexity of morphemes, and the syntactic ordering of subjects, objects, and verbs. To mitigate this, more NL-FOL datasets in a greater variety of languages are needed. A notable instance of such attempted mitigation was the work of researchers who implemented two mitigations to this problem Lu et al. (2022). Firstly, they constructed a Chinese FOL dataset. Secondly, and equally promising, used reinforcement learning based approaches and score-based reward models in the absence of ground-truth NL-FOL datasets. More research in these directions would greatly aid in mitigating this concern.

References

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, Lucy Sun, Alex Wardle-Solano, Hannah Szabo, Ekaterina Zubova, Matthew Burtell, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, Ansong Ni, Linyong Nan, Jungo Kasai, Tao Yu, Rui Zhang, Alexander R. Fabbri, Wojciech Kryscinski, Semih Yavuz, Ye Liu, Xi Victoria Lin, Shafiq Joty, Yingbo Zhou, Caiming Xiong, Rex Ying, Arman Cohan, and Dragomir Radev. 2024. Folio: Natural language reasoning with first-order logic.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes.

Xuantao Lu, Jingping Liu, Zhouhong Gu, Hanwen Tong, Chenhao Xie, Junyang Huang, Yanghua Xiao, and Wenguang Wang. 2022. Parsing natural language into propositional and first-order logic with dual reinforcement learning. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5419–5431, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Wang. 2023. Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association* *for Computational Linguistics: EMNLP 2023*, pages 3806–3824, Singapore. Association for Computational Linguistics.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. 2023. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, Toronto, Canada. Association for Computational Linguistics.

Jidong Tian, Yitian Li, Wenqing Chen, Liqiang Xiao, Hao He, and Yaohui Jin. 2021. Diagnosing the first-order logical reasoning ability through logicnli. pages 3738–3747.

Peiyi Wang, Lei Li, Liang Chen, Feifan Song, Binghuai Lin, Yunbo Cao, Tianyu Liu, and Zhifang Sui. 2023. Making large language models better reasoners with alignment.

Yuan Yang, Siheng Xiong, Ali Payani, Ehsan Shareghi, and Faramarz Fekri. 2023. Harnessing the power of large language models for natural language to first-order logic translation.

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

A Appendix (optional)

A.1 Class A Rationale Generation Prompt

Prompt 1:
Here is a natural language statement: **nl** Here is its first order logic translation: **fol** Your task is to write a brief explanation to teach how to come up with the first order logic (FOL) statement when given the natural language (NL) statement. Be brief. Actually explain how or why the logic works, why those specific logical operators were chosen, how to identify what are the predicates and its arguments, etc. Explain the reasoning behind these choices, don't just state them. When explaining the reasoning, quote evidence for the reasoning directly from the natural language statement and quote words directly corresponding to choice of logical operators. REFERENCING THE NATURAL LANGUAGE STATEMENT WHEN EXPLAINING IS VERY VERY VERY IMPORTANT. Strictly follow the same format exactly as the example outputs. Provide the explanation in a similar way with references from the natural language statement. Only output the explanation, nothing else, no other comment or remarks, and no other headings. Follow the same subheadings as in the example outputs. 3 EXAMPLES EXAMPLE INPUT 1 NL: An architecturally interesting building features unique design elements, visually appealing materials, and innovative construction techniques. FOL: x (Building(x) UniqueDesignElements(x) VisuallyAppealingMaterials(x) InnovativeConstructionTechniques(x) $\rightarrow$ ArchitecturallyInteresting(x)) EXAMPLE OUTPUT 1: Identify Key Entities and Relationships: The key entities are Building, UniqueDesignElements, VisuallyAppealingMaterials, InnovativeConstructionTechniques, and ArchitecturallyInteresting. The relationships are described as attributes of the building. Use variables x for Building. Logical Formulation Structure: x (Building(x) UniqueDesignElements(x) VisuallyAppealingMaterials(x) InnovativeConstructionTechniques(x) $\rightarrow$ ArchitecturallyInteresting(x)) Explanation for Logical Formulation: The universal quantifier $\forall x$ ensures the statement applies to all buildings ("An architecturally interesting building"). The predicates Building(x), UniqueDesignElements(x), VisuallyAppealingMaterials(x), and InnovativeConstructionTechniques(x) identify the building and its attributes. The implication ($\rightarrow$) indicates that if all these attributes are true for a building, then it is architecturally interesting, capturing the condition "features unique design elements, visually appealing materials, and innovative construction techniques" leading to "architecturally interesting". Logical conjunctions ($\wedge$) link the attributes ("unique design elements", "visually appealing materials", and "innovative construction techniques"). EXAMPLE INPUT 2 NL: A rainbow appears when sunlight is refracted, reflected, and dispersed by water droplets in the atmosphere. FOL: x (Rainbow(x) y z (Sunlight(y) WaterDroplets(z) Refracted(y, z) Reflected(y, z) Dispersed(y, z))) EXAMPLE OUTPUT 2: Identify Key Entities and Relationships: The key entities are Rainbow, Sunlight, and WaterDroplets, and the relationships are Refracted (Sunlight, WaterDroplets), Reflected (Sunlight, WaterDroplets), and Dispersed (Sunlight, WaterDroplets). Use

variables x for Rainbow, y for Sunlight, and z for WaterDroplets. Logical Formulation Structure: x (Rainbow(x) y z (Sunlight(y) WaterDroplets(z) Refracted(y, z) Reflected(y, z) Dispersed(y, z))) Explanation for Logical Formulation: The universal quantifier $\forall x$ ensures the statement applies to all instances where a rainbow appears ("A rainbow appears"). Existential quantifiers $\exists y$ and $\exists z$ indicate there exists sunlight and water droplets necessary for the appearance of a rainbow. The predicates Rainbow(x), Sunlight(y), and WaterDroplets(z) identify the rainbow, sunlight, and water droplets respectively. Refracted(y, z), Reflected(y, z), and Dispersed(y, z) describe the physical interactions between sunlight and water droplets. The biconditional operator $\leftrightarrow$ in $(Rainbow(x) \leftrightarrow ...)$ specifies that a rainbow's appearance is directly and exclusively tied to the conditions of sunlight being refracted, reflected, and dispersed by water droplets, capturing the entirety of the condition "when sunlight is refracted, reflected, and dispersed by water droplets in the atmosphere". Logical conjunctions ($\wedge$) link these conditions to emphasize that all these interactions are necessary simultaneously for a rainbow to appear. EXAMPLE INPUT 3 NL: A cake is ready to be served if it's cooled and the frosting is applied. FOL: ReadyToServe(cake) (Cooled(cake) FrostingApplied(cake)) EXAMPLE OUTPUT 3: Identify Key Entities and Relationships: The key entities are "cake," and the states or conditions affecting it are represented by the predicates "Cooled" and "FrostingApplied." The primary relationship described is "ReadyToServe," indicating the cake's readiness based on certain conditions. Logical Formulation Structure: ReadyToServe(cake) (Cooled(cake) FrostingApplied(cake)) Explanation for Logical Formulation: The biconditional operator () is used to express that the readiness of the cake to be served is directly and exclusively linked to two simultaneous conditions: the cake being cooled and having frosting applied. This operator choice directly corresponds to the phrasing "A cake is ready to be served if it's cooled and the frosting is applied," which indicates a necessary and sufficient condition relationship. The logical conjunction ( ) connects "Cooled(cake)" and "FrostingApplied(cake)" to reflect the conjunction "and" in the natural language statement, showing that both conditions must be simultaneously true for the cake to be considered ready to serve. The predicates "Cooled" and "FrostingApplied" clearly encapsulate the conditions mentioned in the statement, identifying the specific states or actions required for the cake to be ready.

Prompt 2: Here is a natural language statement: **nl** Here is its first order logic translation: **fol** Your task is to write a brief explanation to teach how to come up with the first order logic (FOL) statement when given the natural language (NL) statement. Be brief. Actually explain how or why the logic works, why those specific logical operators were chosen, how to identify what are the predicates and its arguments, etc. Explain the reasoning behind these choices, don't just state them. When explaining the reasoning, quote evidence for the reasoning directly from the natural language statement and quote words directly corresponding to choice of logical operators. REFERENCING THE NATURAL LANGUAGE STATEMENT WHEN EXPLAINING IS VERY VERY VERY IMPORTANT. Strictly follow the same format exactly as the example outputs. Provide the explanation in a similar way with references from the natural language statement. Only output the explanation, nothing else, no other comment or remarks, and no other headings. Follow the same subheadings as in the example outputs. 3 EXAMPLES EXAMPLE INPUT 1 NL: A coat provides warmth, an umbrella protects from rain, and sunglasses shield from sunlight. FOL: x y z (Coat(x) $\rightarrow$ ProvidesWarmth(x)) (Umbrella(y) $\rightarrow$ ProtectsFromRain(y)) (Sunglasses(z) $\rightarrow$ ShieldsFromSunlight(z)) EXAMPLE OUTPUT 1: Identify Key Entities and Relationships: The key entities are Coat, Umbrella, and Sunglasses, and the relationships are ProvidesWarmth (Coat), ProtectsFromRain (Umbrella), and ShieldsFromSunlight (Sunglasses). Use variables x for Coat, y for Umbrella, and z for Sunglasses. Logical Formulation Structure: x y z (Coat(x) $\rightarrow$ ProvidesWarmth(x)) (Umbrella(y) $\rightarrow$ ProtectsFromRain(y)) (Sunglasses(z) $\rightarrow$ ShieldsFromSunlight(z)) Explanation for Logical Formulation: The universal quantifiers $\forall x$, $\forall y$, and $\forall z$ ensure the statement applies to all instances of coats, umbrellas, and sunglasses respectively. This captures the generality of the statement "A coat", "an umbrella", and "sunglasses" in the natural language statement. The predicates Coat(x), Umbrella(y), and Sunglasses(z) identify each item as belonging to their respective categories. The implications in the form (Coat(x) $\rightarrow$ ProvidesWarmth(x)), (Umbrella(y) $\rightarrow$ ProtectsFromRain(y)), and (Sunglasses(z) $\rightarrow$ ShieldsFromSunlight(z)) express the functional properties of each item as stated directly in the natural language: a coat "provides warmth", an umbrella "protects from rain", and sunglasses "shield from sunlight". Logical conjunctions ($\wedge$) connect these independent clauses, reflecting the structured listing of item functions in the natural language statement. EXAMPLE INPUT 2 NL: Hospitals treat patients with medical professionals and appropriate equipment. FOL:

xy (Hospital(x) Patient(y) → zw (MedicalProfessional(z) Equipment(w) TreatsWith(x, y, z, w))) EXAMPLE OUTPUT 2: Identify Key Entities and Relationships: The key entities in the natural language statement are "Hospitals", "patients", "medical professionals", and "equipment". The relationship expressed is "treats", involving hospitals, patients, medical professionals, and equipment. We use variables x for Hospitals, y for Patients, z for Medical Professionals, and w for Equipment. Logical Formulation Structure: $\forall x \forall y (Hospital(x) \wedge Patient(y) \rightarrow \exists z \exists w (MedicalProfessional(z) \wedge Equipment(w) \wedge TreatsWith(x, y, z, w)))$ Explanation for Logical Formulation: The universal quantifiers $\forall x$ and $\forall y$ ensure the statement applies to all hospitals and all patients, reflecting the generality suggested by the phrase "Hospitals treat patients". The existential quantifiers $\exists z$ and $\exists w$ indicate there exists at least one medical professional and one piece of equipment involved in the treatment, aligning with the inclusion of "medical professionals and appropriate equipment". The predicates Hospital(x), Patient(y), MedicalProfessional(z), and Equipment(w) define the entities involved. The predicate TreatsWith(x, y, z, w) describes the treatment relationship, capturing the action "treat" specified in the natural language. Logical conjunctions ($\wedge$) connect the characteristics of each entity, and the implication ($\rightarrow$) in the formula represents the conditional nature of the treatment process stated as "Hospitals treat patients with medical professionals and appropriate equipment". This logical operator choice ensures that for each hospital and patient, if they are indeed a hospital and a patient, there must exist some medical professional and equipment that participate in the treatment process. EXAMPLE INPUT 3 NL: A dessert is delicious if it's sweet but not overly sweet and has a pleasant texture. FOL: DeliciousDessert(x) (Sweet(x) ¬OverlySweet(x) PleasantTexture(x)) EXAMPLE OUTPUT 3: Identify Key Entities and Relationships: The key entities are "dessert" and the qualitative attributes related to it: "sweet," "overly sweet," and "pleasant texture." The relationships captured are conditions that determine whether a dessert is considered delicious. The variable x represents a dessert. Logical Formulation Structure: DeliciousDessert(x) (Sweet(x) ¬OverlySweet(x) PleasantTexture(x)) Explanation for Logical Formulation: The biconditional operator is used to express that a dessert being delicious is precisely equivalent to the conditions specified. The statement "A dessert is delicious if it's sweet but not overly sweet and has a pleasant texture" directly suggests this equivalence. The conjunction $\wedge$ connects the predicates Sweet(x), ¬OverlySweet(x), and PleasantTexture(x), reflecting the conjunction "and" used in the natural language to cumulatively describe the conditions for a dessert being delicious. The predicate Sweet(x) directly maps to the word "sweet," ¬OverlySweet(x) (where ¬ represents negation) captures "not overly sweet," addressing the negation "not" used in the statement. Finally, PleasantTexture(x) corresponds to "has a pleasant texture," aligning directly with these words. The use of these logical operators and predicates accurately encodes the conditional and exclusionary nuances from the natural language description into the first order logic expression.

A.2 Class B Rationale Generation Prompt

Here is a natural language statement: **nl** Here is its first order logic translation: **fol**

Your task is to write a brief explanation to teach how to come up with the first order logic (FOL) statement when given the natural language (NL) statement. Be brief. Actually explain how or why the logic works, why those specific logical operators were chosen (including operators like not, implication, exclusive or, etc.), how to identify what are the predicates and its arguments, how to decide what the quantifiers are, etc. Explain the reasoning behind these choices, don't just state them. When explaining the reasoning, quote evidence for the reasoning directly from the natural language statement. REFERENCING THE NATURAL LANGUAGE STATEMENT WHEN EXPLAINING IS VERY VERY VERY IMPORTANT.

Strictly follow the same format exactly. Provide the explanation in a similar way with references from the natural language statement. Only output the explanation, nothing else, no other comment or remarks, and no other headings. Nothing except the 3 points in the explanation. DO NOT INCLUDE the title "Explanation".

EXAMPLE

NL: A vending machine dispenses items such as snacks or beverages in exchange for payment, often in the form of coins or bills.

FOL: x y z (VendingMachine(x) Item(y) Payment(z) → (Dispenses(x, y) InExchangeFor(x, z) (AcceptsCoins(x) AcceptsBills(x))))

Explanation: - Identify Key Entities and Relationships: The key entities are VendingMachine, Item, and Payment (with subtypes Coin and Bill), and the relationships are Dispenses (VendingMachine, Item), InExchangeFor (Item, Payment), AcceptsCoins (VendingMachine), and AcceptsBills (VendingMachine). Use variables x for VendingMachine, y for Item, and z for Payment. - Logical Formulation Structure: Formulate the logic: $\forall x \exists y \exists z$ (VendingMachine(x) ∧ Item(y) ∧ Payment(z) → (Dispenses(x, y) ∧ InExchangeFor(x, z) ∧ (AcceptsCoins(x) ∨ AcceptsBills(x))). - Explanation for Logical Formulation: The universal quantifier $\forall x$ ensures the statement applies to all vending machines ("A vending machine"). Existential quantifiers $\exists y$ and $\exists z$ indicate there exists an item and a payment. The predicates VendingMachine(x), Item(y), and Payment(z) identify the vending machine, item, and payment respectively. Dispenses(x, y) and InExchangeFor(x, z) describe the relationships of dispensing items and receiving payments. The disjunction (∨) in $(AcceptsCoins(x) \vee AcceptsBills(x))$ specifies that the vending machine can accept either coins or bills, capturing the condition "often in the form of coins or bills". Logical conjunctions (∧) link conditions ("dispenses items...in exchange for payment").