A Study on LLM Autoformalization and Symbolic Reasoning Using First-Order Logic

Anonymous ACL submission

Abstract

We present a three-step evaluation pipeline for formal reasoning in First-Order Logic (FOL), covering NL→FOL translation, neuro-symbolic inference, and FOL→NL back-translation. We introduce LogicSim+, an extension of the original LogicSim metric that corrects mathematical inconsistencies, integrates Prover9 for syntactic validation, and extends evaluation to back-translation. We also release a corrected version of the FOLIO dataset, fixing 3.24% of malformed entries. Three strategies are evaluated across 10 open-source LLMs (1.5B–20B parameters): one-shot prompting, syntax-restricted generation via context-free grammars, and KTO-based preference alignment. Results show that while models extract semantic relationships effectively, maintaining logical consistency in FOL remains a challenge. Syntax-restricted generation improves semantic similarity, and alignment produces mixed effects depending on the model and task. Inference performance improves with model size for reasoning models, with the best model approaching the performance of Logic-LM++. Our findings highlight that autoformalization and neuro-symbolic reasoning are complementary but distinct capabilities in modern LLMs.

1 Introduction

Reasoning has been a core part of Large Language Model (LLM) evaluation in recent years. State-of-the-art post-training methodologies, such as alignment (Shao et al., 2024; Ahmadian et al., 2024), have improved LLM performance on mathematical benchmarks like GSM8K[1] (Cobbe et al., 2021), MATH[2] (Hendrycks et al., 2021), and AIME[3] (Dekoninck et al., 2026). These benchmarks focus on general mathematics, where the answers are limited to numerical or algebraic expressions. However, formal mathematical reasoning is not restricted to single answer questions. Logic, vital for formal reasoning, is largely absent from standard LLM benchmarks.

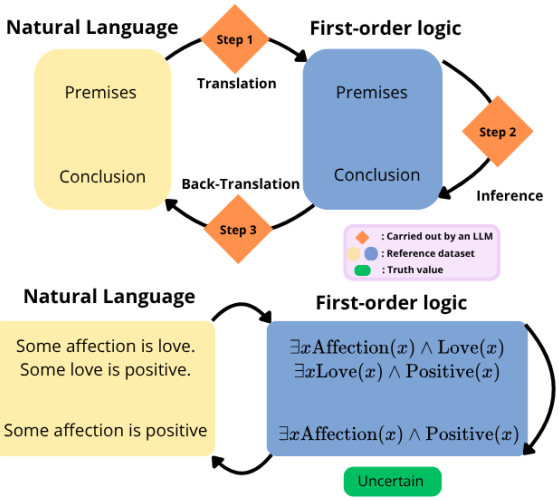

Figure 1: **Left**: Pipeline Schema. **Right**: Natural Language and FOL example. Green text corresponds to the inference value of the FOL conclusion.

Measuring reasoning based solely on incomplete benchmarks leads to an incomplete evaluation of LLMs' mathematical capabilities. To fully evaluate reasoning abilities, two additional tasks must be included: Autoformalization (automatic translation of mathematical content from natural language to a formal symbolic expression (Zhang et al., 2024; Wang et al., 2018; Szegedy, 2020)) and Symbolic Reasoning (logically valid inference procedures based on expressions in a formal language).

Our main contributions are as follows:

- We propose a three step pipeline that evaluates autoformalization: NL→FOL translation, neuro-symbolic inference, and FOL→NL back-translation (See Figure 1).
- We introduce *LogicSim+*, an extension of the

[1] https://huggingface.co/datasets/openai/gsm8k
[2] https://huggingface.co/datasets/nlile/hendrycks-MATH-benchmark
[3] https://huggingface.co/datasets/MathArena/aime_2025

original metric with mathematical corrections, external prover integration for logical validity evaluation, and support for back translation.

- We release a corrected version of FOLIO, fixing 3.24% items with minimal edits.
- We compare three different strategies: one-shot prompting, syntax constraint generation and logic based alignment.
- We systematically evaluate 10 open-source LLMs of different architecture families and sizes, providing a per-step analysis.

2 Related Work

Autoformalization and Formal Reasoning Autoformalization is the task of translating a problem in natural language (NL) to a formal representation capable of formal mathematical proofs (Szegedy, 2020). Early work in this field focused on syntax compatibility with the proof assistant Isabelle (Wu et al., 2022). As LLMs have improved, so has performance and flexibility of complex theorem proving pipelines (Ranaldi et al., 2025; Quan et al., 2025; Xu et al., 2026). More recent work has moved towards proof assistants like Lean (Yang et al., 2023; Hsiang et al., 2025; Shen et al., 2025), with most works evaluating systems in the MiniF2F and ProofNet benchmarks (Zheng et al., 2022; Azerbayev et al., 2023; Weng et al., 2025). Even if these benchmarks are supported by an automated theorem prover, such support guarantees formalization but not an actual representation of the problem at hand (Ammanamanchi and Bhat, 2025). In addition, these datasets do not integrate formal logical reasoning, leaving out an important area of mathematical decision making.

FOL Datasets Unlike proof-assistant benchmarks, some datasets are designed to evaluate both autoformalization and formal logical reasoning. ProofWriter (Tafjord et al., 2021), PrOntoQA (Saparov and He, 2023), MALLS (Yang et al., 2024) and Willow (Deveci, 2024) are datasets that serve as a benchmark for autoformalization and formal reasoning. However, some either lack inference related evaluation or are partially curated and generated using LLMs. FOLIO (Han et al., 2024) is a human-curated dataset that goes beyond autoformalization and contains neuro-symbolic inference values, enabling an in-depth analysis of various aspects of logical reasoning, while avoiding circular evaluation of language models.

FOL Evaluation Based on such datasets, works like Logic-LM (Pan et al., 2023), LINC (Olausson et al., 2023), and Logic-LM++ (Kirtania et al., 2024) propose a pipeline that tests LLMs' inference capabilities with the help of external provers. While these works tackle autoformalization and neuro-symbolic reasoning in FOL, performance is only measured using accuracy of a model's truth value prediction rather than with any sort of intermediate translation value.

Such intermediate evaluations complement truth value predictions and allow a fine grained analysis throughout the reasoning pipeline. Character matching (Levkovskyi and Li, 2021) is a simple yet very harsh evaluation for this task since it punishes lexical and syntactical variations. Logical Equivalence (Yang et al., 2024) obtains boolean validity between sets of premises, however it disregards the values of quantifiers and constants. LogicSim (López-Ponce and Bel-Enguix, 2025) is a performance indicator that mixes logical and semantic similarity throughout the autoformalization process.

LLM Alignment Post-training alignment aims to steer model behavior, making it a natural candidate for improving structured generation tasks like autoformalization. Particularly, Reinforcement Learning from Human Feedback (RLHF) has proven to be effective at enhancing generative capabilities (Ziegler et al., 2020; Ouyang et al., 2022), as well as following safety guidelines (Lu et al., 2025). These methods guide a model adjust model outputs by assigning scalar rewards to generations associated to a prompt, guiding the model towards preferred responses. Algorithms such as DPO (Rafailov et al., 2023) and KTO (Ethayarajh et al., 2024) set such rewards via human preferences, either continuous or binary, over pairs of responses.

3 Problem Definition

The experimental pipeline consists of three subtasks implemented and evaluated independently: 1. Translation 2. Inference 3. Back-translation. Figure 1 shows an abstract example of the pipeline. Consider the following values:

- P^{NL}_{gold}: a set of NL premises extracted from a reference dataset
- P^{FOL}_{gold}: a set of FOL premises extracted from a reference dataset

- P^{FOL}_{llm}: a set of FOL premises generated by an LLM
- C^{FOL}_{gold}: a single FOL conclusion extracted from a reference dataset
- C^{NL}_{gold}: a single NL conclusion extracted from a reference dataset
- C^{NL}_{llm}: a single NL conclusion generated by an LLM

Then the truth value of the query: can C^{NL}_{gold} be inferred from P^{FOL}_{gold}? is denoted as $P^{FOL}_{gold} \models C^{NL}_{gold}$.

Translation An LLM φ transforms NL premises into their corresponding FOL expressions:

$$\varphi(P^{NL}_{gold}) = P^{FOL}_{llm} \tag{1}$$

Performance is measured by $LogicSim +_t (P^{FOL}_{llm}, P^{FOL}_{gold})$.

Inference An LLM carries out neuro-symbolic reasoning to determine if a single conclusion can be inferred from a set of premises:

$$\delta = \varphi(P^{FOL}_{gold} \models C^{NL}_{gold}) \tag{2}$$

With $\delta \in \{True, False, Uncertain\}$. Performance is measured using accuracy over all predictions.

Back-translation An LLM expresses a single FOL conclusion in NL:

$$\varphi(C^{FOL}_{gold}) = C^{NL}_{llm} \tag{3}$$

Performance is measured by $LogicSim +_b (C^{FOL}_{gold}, C^{NL}_{llm})$.

4 LogicSim+

We present LogicSim+, an extension and improvement over the original LogicSim (López-Ponce and Bel-Enguix, 2025). The original metric sums the differences of logically relevant values: total premise difference (pd), total and distinct predicate differences (ap, tp), total logical operators difference (ld), as well as the intersection-over-union (IoU) of predicates:

$$LogicSim = pd + ap + tp + ld + IoU \tag{4}$$

LogicSim+ addresses three issues with this metric across the full pipeline: corrects mathematical inconsistencies in the translation step, extends evaluation to back-translation, and integrates an external prover for inference. Figure 2 shows an example of LogicSim+ for steps 1 and 3.

Translation Two corrections are applied to LogicSim in the translation step. The first correction is the removal of the IoU component. Differences in predicate and variable count imply that a larger value signifies a larger difference between each premise, however IoU is a similarity coefficient hence a higher value implies higher similarity. In addition, the other differences are unbound integer values, whereas $IoU \in [0, 1]$ for any two premises. Removing these values makes LogicSim+ easier to interpret, and allows unbounded values that are key for a discrete similarity value.

The second correction extends $\Delta p(x, y) = tp + ap$ to both variables and logical operators. Measuring both total and distinct counts is key to understanding model consistency during translation. Thus we define $\Delta v(x, y)$ and $\Delta l(x, y)$, extending ld to both total and distinct counts. The translation step of LogicSim+, denoted by $LogicSim{+}_t$, is defined in equation 5.

$$LogicSim{+}_t = pd + \Delta p + \Delta v + \Delta l \tag{5}$$

Back-translation LogicSim+ extends evaluation to the back-translation step. A consistent back-translation preserves constant and predicate references as well as the total count of logical operators. This preserves the logical structure of the premise even if the semantics of the translation differ.

In order to measure this consistency, we match logical elements of a FOL premise with semantic elements of a NL sentence. Logical operators ($\forall, \exists, \vee, \wedge, \neg, \oplus$) are matched with their corresponding NL expression (e.g. $\wedge$ with 'and'), whereas predicates and variables are matched with nouns, verbs, and adjectives. We denote $|C|$ as the total count of variables, predicates, and logical operators in the conclusion C. We define $LogicSim{+}_b$ in equation 6.

$$LogicSim{+}_b = |C^{FOL}_{gold}| - |C^{NL}_{llm}| \tag{6}$$

A back-translation can be compressed, extended or exact, depending on how a model translates a premise. A **compressed** translation has less logical values in NL than in FOL ($LogicSim{+}_b > 0$), while an **extended** translation has more values in NL than in the original FOL form ($LogicSim{+}_b < 0$). An **exact** translation contains the same number of elements in FOL as in NL ($LogicSim{+}_b = 0$). Examples and further discussion can be seen in Appendix D.

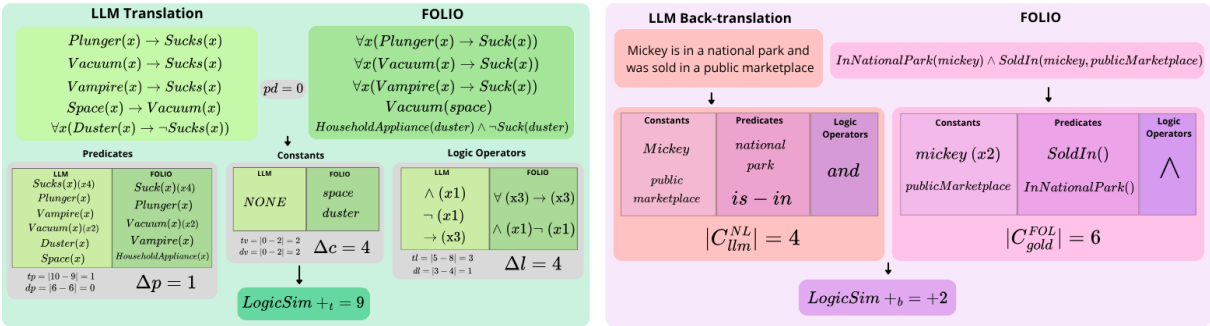

Figure 2: **Left:** An example of $LogicSim{+}_t$. **Right:** An example of $LogicSim{+}_b$.

Inference This step is commonly evaluated using accuracy over truth-value predictions (Pan et al., 2023; Kirtania et al., 2024; Olausson et al., 2023). However, this evaluation does not give an insight into whether a model correctly formalizes the premises it reasons over. We complement it by checking the parsability of LLM-generated premises using Prover9 (McCune, 2005–2010). A pair $(P^{FOL}_{llm}, C^{NL}_{gold})$ is passed to the prover to determine if the conclusion follows from the premises. This pair is automatically translated from symbolic FOL notation into Prover9's FOL syntax without modifying the underlying logical structure.

Since Prover9 lacks the UNCERTAIN output label (32.4% of labels in FOLIO), we shift our focus from the prover's output to whether the premises can be parsed at all. This complements $LogicSim{+}_t$ which only measures lexical differences. Parsing with an external prover adds three things: lexical variation that is logically irrelevant is ignored, syntactical correctness is verified, and the evaluation is LLM-independent. We denote define $LogicSim{+}_{val}$ in equation 7.

$$LogicSim{+}_{val} = \begin{cases} 1 & \text{if P9 parses } P^{FOL}_{llm} \\ 0 & \text{elsewhere} \end{cases} \quad (7)$$

5 Methodology

We evaluate three strategies: one-shot in-context learning as a baseline, syntax-restricted generation, and KTO alignment. We invesigate whether alignment can improve formalization performance, and whether restricting generation to a syntactically defined structure helps with the structured nature of FOL outputs. Technical details regarding hardware and hyperparameters can be found in Appendix B.

5.1 One-Shot Learning

Baseline prompts follow a one-shot format, with a structure similar to that of Logic-LM (Pan et al., 2023). Regardless of the step, they include a description of the problem as well as what constitutes a solution, a single example of the required behavior, and the actual instance needed to evaluate. Prompts can be found in Appendix A.

5.2 Syntax Restricted Generation

Syntax errors are a bottleneck for autoformalization (Tam et al., 2024). Even if logical relationships are correctly extracted, they have to be expressed under a particular syntax, especially when integrating external provers. To handle this, we limit an LLM's generation using a context-free grammar (CFG) that models FOL. This restriction is only applicable to step 1 of the pipeline. Evaluation with omit steps 2, 3, and perplexity scores.

We propose two CFGs based on Lark. The first one parses whitespace and various Unicode representations for logical symbols in the FOLIO dataset, thus keeping the modifications to the base dataset minimal. However, during generation we found that whitespace and Unicode representations of logical symbols introduce token variations, where different symbols of the same operators result in different tokenization. To mitigate this, we modify the base grammar to remove whitespace and handle Unicode variations of logical operators. The full CFG can be seen in Appendix F.

5.3 Alignment

For alignment, we choose KTO (Ethayarajh et al., 2024) as a preference-based approach. KTO is better suited for this task over the popular DPO since it uses binary labels of preferred vs rejected responses, avoiding the need for preference scores over pairs. Hyperparameter details can be found in Appendix E.1.

5.4 Answer Filtering

While most generations follow structures that can be automatically filtered, inconsistent patterns arise that have to be manually extracted. Filtering is done using regular expressions, Python's text manipulation functions, and manual selection over specific pathological responses. We release both raw and the filtered answers. For further discussion refer to Appendix C.1 and C.2.

6 Experimental Setup

6.1 Models

We experiment with a variety of open-source models, ranging from small to medium size (1.5B-20B). The selected model families are Qwen3 (Team, 2025b), Qwen3.5 (Qwen Team, 2026), Gemma 3 (Team, 2025a), gpt-oss (OpenAI, 2025), and DeepSeek R1 (distilled versions) (DeepSeek-AI, 2025). A quantized version of each model was used whenever available. Model abbreviations used throughout are: Q3 (Qwen3), Q3.5 (Qwen3.5), G3 (Gemma 3), GPT (gpt-oss), DS-0528 and DS-DIST (DeepSeek R1 distilled versions), followed by parameter count where applicable. The full model list is shown in Table 1. Additional discussion regarding the selection of evaluated models can be found in Appendix C.2.

6.2 Evaluation Dataset

We use the FOLIO dataset for evaluation. While parsing it using our grammar, we found that the parse rate over the FOL expressions was 96.76%. We apply syntax correction to the 3.24% wrongly parsed formulas, limiting edits to fixing unbalanced parentheses, missing commas, quantifiers with no variable, and similar surface-level errors. Tables 7 and 8 show parsing error results.

6.3 Alignment Dataset

We create an alignment dataset for KTO training. The algorithm requires prompts paired with preferred and rejected responses, and binary labels indicating preferences. Preferred responses are gold answers extracted from the FOLIO dataset. Rejected responses are LLM-generated answers for the same prompt. The result is a collection of triplets (p_i, a_i, ℓ_i) where p_i corresponds to a one-shot prompt, a_i is the corresponding FOLIO answer, and $\ell_i \in \{\text{TRUE}, \text{FALSE}\}$ is the binary preference label. The dataset consists of 6006 triplets. Since each model is aligned on its own generations, model-specific versions of the dataset are released via HuggingFace[4]. For additional discussion about alignment datasets refer to Appendix E.2.

6.4 Evaluation Metrics

As the main metric, we use LogicSim+. Step 2 (Inference) is also evaluated using accuracy over label prediction. Additionally, we measure a generation average perplexity for each step of the pipeline. Perplexity gives an insight into how confident each model is at inference time. By contrasting this value with the rest of the scores, it is possible to analyze both LogicSim+ as a performance indicator, and how well an LLM grounds its answers in a given task.

7 Results

Results are shown in Table 1. Scores are reported as (Validation, Test) unless stated otherwise. Step 1 and 3 show average score per split.

7.1 Pipeline

Translation Models struggle when dealing with multiple quantifiers within a single premise. In the example below, DS-0528 drops the second quantifier and compensates by adding an extra predicate, making the premise syntactically invalid.

- $\forall x \forall y$ (SuperheroMovie(x) $\wedge$ NamedAfter(x, y) $\rightarrow$ GoodGuy(y)) (FOLIO)
- $\forall x$ (SuperheroMovie(x) $\wedge$ NamedAfterCharacter(x)) $\rightarrow$ GoodGuy(character(x)) (DS-0528)

Models are also prone to syntactical errors. Prover9 only accepts (60.4%, 70.4%) of premises. Arity errors occur, on average, in (6.9%, 9.5%) of instances, syntax errors in (30.6%, 20.5%) in baseline conditions. Table 9 contains details on error types.

Inference Models struggle the most on the inference step. Only 4 of 10 models (Q3.5-4B, Q3.5-9B, G3-4B, Q3-14B) produce valid labels for every instance. The rest fall into thinking loops or pathological generations at least once. DS-DIST is the worst model with 61 degenerate instances. Table 9 shows the full count of incorrect generations. Reported scores omit incorrectly generated labels.

However, as parameter size increases, reasoning models consistently predict over 70% of labels. The best model outperforms Logic-LM (78.92%)

[4]REDACTED FOR ANONYMOUS SUBMISSION

Strategy	Model	$LS+_t$	Acc (%)	$LS+_{val}$	$LS+_b$	Perplexity
One-Shot	Q3-4B	(24.54, 29.07)	(53.7, 51.3)	(0.71, 0.69)	(-0.37, -0.62)	(1.016, 1.008)
	Q3-8B	(22.33, 23.71)	(61.6, 58.4)	(0.72, 0.73)	(**0.01**, -0.48)	(1.027, 1.018)
	Q3-14B	(21.81, 24.35)	(**82.8**, **74.3**)	(0.71, 0.73)	(0.02, -0.41)	(1.073, 1.071)
	Q3.5-4B	(21.96, 22.41)	(52.7, 51.8)	(0.67, 0.69)	(-0.6, -1.31)	(1.202, 1.199)
	Q3.5-9B	(21.12, 22.8)	(55.2, 55.4)	(0.64, 0.64)	(-0.56, -1.24)	(1.049, 1.049)
	G3-4B	(24.28, 24.68)	(41.9, 40.7)	(0.46, 0.58)	(-0.43, -0.92)	(1.054, 1.055)
	G3-12B	(**19.27**, **20.7**)	(52.7, 53.5)	(0.71, 0.68)	(-0.4, -0.62)	(1.053, 1.06)
	GPT-20B	(26.36, 25.72)	(73.4, 65.9)	(**0.73**, **0.88**)	(-0.03, **-0.31**)	(1.297, 1.291)
	DS-0528	(23.2, 24.91)	(71.4, 63.7)	(0.6, 0.69)	(-0.09, -0.45)	(1.096, 1.093)
	DS-DIST	(47.24, 36.23)	(47.3, 43.4)	(0.66, 0.73)	(-0.19, -0.62)	(1.101, 1.105)
Syntax	Q3-4B	(18.81, 24.2)	-	(0.59, 0.52)	-	-
	Q3-8B	(17.44, 22.14)	-	(0.61, 0.57)	-	-
	Q3-14B	(17.13, 19.62)	-	(0.38, 0.4)	-	-
	Q3.5-4B	(18.58, 19.18)	-	(0.63, 0.62)	-	-
	Q3.5-9B	(18.73, 18.18)	-	(0.6, 0.54)	-	-
	G3-4B	(19.80, 21.05)	-	(0.57, 0.5)	-	-
	G3-12B	(17.25, 17.73)	-	(0.71, 0.71)	-	-
	GPT-20B	(14.9, 15.32)	-	(**0.74**, **0.75**)	-	-
	DS-0528	(50.77, 49.3)	-	(0.45, 0.64)	-	-
	DS-DIST	(**10.44**, **12.21**)	-	(0.73, 0.73)	-	-
KTO	Q3-14B	(**20.14**, 20.26)	(**59.1**, **58.4**)	(**0.75**, **0.79**)	(0.28, -0.1)	(0.038, 0.038)
	G3-12B	(21.1, **14.0**)	(45.3, 50.4)	(0.75, 0.63)	(**0.06**, 0.8)	(0.041, 0.041)
	DS-0528	(43.14, 32.87)	(30.5, 27)	(0.59, 0.69)	(-37.28, **0.03**)	(0.03, 0.032)

Table 1: Results across all strategies. Coordinate pairs show (validation, test). Model abbreviations: Q3 = QWEN3, Q3.5 = QWEN3.5, G3 = GEMMA-3-IT, GPT-20B = GPT-OSS-20B, DS-0528 = DEEPSEEK-R1-0528-QWEN3-8B, DS-Dist = DEEPSEEK-R1-DISTILL-QWEN-7B. Bold values highlight best performing model per strategy. Val-Test dataset size: 203, 226.

and LINC (72.5%), and stays just below of Logic-LM++ (84.4%).

Back-translation Models tend to back-translate without modifying the original structure. For conclusions with few logical values (< 5), $LogicSim+_b$ stays within 0 ± 2. Even for complex ones (> 15) the results stay within a 0 ± 3. The following example shows both compressed and extended translations within this range.

- (¬ExerciseEveryDay(james) ∧ Play(james, division1Football) ∧ PlayAt(james, yaleUniversity)) → (Play(james, division1Football) ∧ PlayAt(james, yaleUniversity)) (FOLIO)
- *If James does not exercise every day and plays division 1 football and plays at Yale University, then James plays division 1 football and plays at Yale University.* (GPT-20B, $LS+_b = -3$)
- *If James does not exercise every day and plays Division I football at Yale University, then he plays Division I football at Yale University.* (DS-0528, $LS+_b = 2$)

Models become volatile with multiple quantifiers, as shown in the example below where $LogicSim+_b$ ranges from -6 to 7.

- ∀x ∀y ((Roof(x) ∧ Concrete(x) ∧ Roof(y) ∧ Seagrass(y)) → Stronger(x, y))
- *Every concrete roof is stronger than every seagrass roof.* (GPT-OSS $LS+_b = 7$)
- *For all x and y, if x has a roof and is made of concrete,*

and y has a roof and is made of seagrass, then x is stronger than y. (Q3.5-9B $LS{+}_b = -6$)

7.2 Incorrect notation

Results so far have focused on syntactically correct answers. However, models do not always produce answers with such structure, or even FOL symbols. Premises containing acronyms, uncommon predicates/constants (3 or more words, or specific proper nouns), or numerical related reasoning lead to answers using mathematical notation not present in FOL ($\in, \equiv, \{, \}$), misplaced truth values, or plain NL copies. Examples are shown below.

- Reading $\in$ IELTS, Listening $\in$ IELTS, Writing $\in$ IELTS. (G3-12B)
- Schmelz is a former parade and exercise ground located in Vienna. (Q3-14B)
- 30 $\equiv$ (December 1, 2021 - December 1, 1970) $\mod$ 7 (G3-4B)

7.3 Models

Instruction models The best model for translation is an instruction-tuned model, which is consistent with the token-level precision required for the task. Yet, these models barely surpass a 50% accuracy on the inference and back-translation steps.

Thinking models Thinking models perform similarly in translation regardless of model size. However, this does not hold for the inference and back-translation steps. Thinking models show increased inference performance as model size increases and preserve FOL structure in NL.

Distillation Distilled models are volatile. DS-0528 ranks top-3 across all steps, whereas DS-DIST ranges from bottom-2 to mid. Two additional distilled models were evaluated but their results excluded due to incoherent responses (Appendix C.2).

7.4 LogicSim+

Our metric quantifies model behavior across all steps of the pipeline. $LogicSim{+}_t$ suggests that thinking models translate consistently. Only G3-12B, an instruction-tuned model, stays below a $LogicSim{+}_t$ value of 20. $LogicSim{+}_b$ shows a slight compression tendency, with Q3-8B and Q3-14B being the only models that fully preserve logical elements on average.

7.5 Syntax Restriction

Restricted generation increases semantical similarity and output structure. $LS{+}_t$ drops for all but one model (DS-0528). However syntactical problems do not disappear, and generation is susceptible to loops and out-of-token errors.

7.6 Alignment

KTO has mixed effects in both translation steps. G3-12B improves in autoformalization and logical quality. Q3-14B improves lexical similarity but loses parsing efficiency. Both models shift towards compressed back-translation. DS-0528 is hindered by KTO.

Unsurprisingly, inference performance decreases across all aligned models. Since KTO only signals that the output should be a single truth-value word, reasoning capabilities are suppressed rather than guided.

7.7 Logit Analysis

Models rarely explore next possible tokens (specially aligned models), partially due to generation temperature. However, exploration doesn't imply better performance. The highest exploring models (GPT-20B, Q3.5-4B, DS-DIST) aren't all the best performing ones. In step 2, reasoning models can achieve top results with over 0.2 difference in perplexity from GPT-20B.

8 Discussion

On average, LLMs autoformalize differently from humans. While models extract semantic relationships effectively, representing them in a logically consistent manner remains a challenge. Autoformalization does not translate into neuro-symbolic reasoning, however reasoning is backed by autoformalization. Strong translation models ($LS{+}_{val} > 0.7$) correctly infer anywhere from 53%-83%. Only one model surpasses a 70% test-set accuracy, while three do so in the validation set. The best inference models (Q3-14B, GPT-20B) are backed by strong translators. Back-translation is the most consistent step, with models preserving logical meaning even when simplifying their generations.

The Struggles of Formalizing Autoformalization is a process susceptible to lexical variation. NL constants can be formalized in various manners without impacting logical validity. However, arity consistency or token level operations such as parenthesis matching, key for formal statements,

are areas where LLMs have shortcomings. LLMs preserve predicate/constant naming throughout a set of premises, but remain inconsistent in the actual use of the value (e.g. Premise 1: `winter(feb)`. Premise 2: `feb(now)` (Q3.5-9B)).

Parenthesis matching often occurs with incomplete translations. Notably, it is also related to the use of the XOR operator ($\oplus$). When formalizing arguments containing such operator, LLMs don't associate predicates explicitly (i.e. predicates with structures like $A \vee B \oplus C$ aren't grouped $(A \vee B) \oplus C$ or $A \vee (B \oplus C)$) thus leading to ambiguity in formal interpretation.

Beyond the Prediction Label Models are evaluated by the label they output, however, -THINKING models offer an insight how such final label is produced. The reasoning carried out before outputting an answer is far different from logical deduction. These models fall back into natural language debating and exhaustive analysis of possible logical models that accommodate a particular premise. However, such models are prone to falling in a loop and not being able to output a value. Further examination of such reasoning could lead to a better understanding of an LLM's logical deduction capabilities.

8.1 LogicSim+

By incorporating LogicSim+ into our evaluation we're able to examine particular behavior over the whole pipeline. The integration of logical solvers proves to be a useful mechanism to understand what high and low translation values represent.

Since FOL is subject to lexical variation, if there exists a gold standard, different translations should not be discarded as incorrect. $LogicSim{+}_t$ describes translation behavior as well as logical correctness in FOL.

Back-translation vs Logical Value An important question needs to be asked regarding step 3: Does metric-similarity entail logical similarity? Empirical observation tells us yes, individual quantifier-predicate-constants premises are translated efficiently ($\neg Vegetable(cake)$ becomes `Cake is not a vegetable.`) Composed predicates using OR or AND operators behave similarly.

However, we have to be careful with categorical statements such as these when comparing formal language vs natural language. XOR('A or B, but not both') if often interchanged with OR, with LLMs adding or omitting XOR's latter condition inconsistently.

$\oplus/\vee$ **FOLIO:** HumbleAbout(mary, theirKnowledge) $\vee/\oplus$ (ConfidentAbout(mary, theirKnowledge) **BT:** *Mary is either humble about their knowledge or confident about their knowledge.* (Same translation for different operands) (G3-12B)

9 Conclusions and Future Work

We introduce a three step pipeline for fine grained formal reasoning evaluation in first-order logic. We present LogicSim+, a metric capable of evaluating each step of the pipeline capable of analyzing both semantics and syntax in NL and FOL. We test this pipeline with one-shot learning, syntax restricted generations, and preference-based alignment, using modern LLMs to showcase strengths, limitations, and improvements in logical reasoning capabilities.

While LogicSim+ efficiently evaluates an NL-FOL pipeline, further development matching formal logical structure with its natural language counterpart would benefit back-translation evaluation. Preference-based alignment manages to steer a model's behavior, however this steering isn't suitable for reasoning tasks. Using alignment with verifiable rewards could improve a model's performance in such tasks. In addition, incorporating solvers into these sort of algorithms' reward function could prove to be a key to guaranteeing correct formal statements. Finally, formally comparing logical equivalence of an LLM-generated set of premises with a gold standard can truly show if LLMs are capable of parsable and equivalent autoformalization.

Limitations

LLM Alignment often uses tens of thousands of examples during post-training. However, high quality and human made resources such as FOLIO are scarce. Thus, we either fall into circular reasoning (using LLM-generated content to evaluate an LLM) or have to restrict experiments to smaller post-training datasets.

Manual filtering, while carefully carried out, is subject to human error. Datasets used throughout these experiments were strictly reviewed, yet typos or minor errors could be present in the final versions.

Ethical Considerations

LLMs exhibiting formal reasoning capabilities have potential use in decision making scenarios. Since our results are grounded to verifiable datasets, such use should be constrained since these models are inconsistent and volatile even under these circumstances.

References

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. 2024. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12248–12267, Bangkok, Thailand. Association for Computational Linguistics.

Pawan Sasanka Ammanamanchi and Siddharth Bhat. 2025. Faults in our formal benchmarks. In *The 5th Workshop on Mathematical Reasoning and AI at NeurIPS 2025*.

Zhangir Azerbayev, Bartosz Piotrowski, Hailey Schoelkopf, Edward W. Ayers, Dragomir Radev, and Jeremy Avigad. 2023. Proofnet: Autoformalizing and formally proving undergraduate-level mathematics.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Jasper Dekoninck, Nikola Jovanović, Tim Gehrunger, Kári Rögnvaldsson, Ivo Petrov, Chenhao Sun, and Martin Vechev. 2026. Beyond benchmarks: Matharena as an evaluation platform for mathematics with llms.

Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 2021. 8-bit optimizers via block-wise quantization. *CoRR*, abs/2110.02861.

Tim Dettmers and Luke Zettlemoyer. 2023. The case for 4-bit precision: k-bit inference scaling laws. In *International Conference on Machine Learning*, pages 7750–7774. PMLR.

İbrahim Ethem Deveci. 2024. Transformer models for translating natural language sentences into formal logical expressions. Master's thesis, Middle East Technical University.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Model alignment as prospect theoretic optimization. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, Lucy Sun, Alexander Wardle-Solano, Hannah Szabó, Ekaterina Zubova, Matthew Burtell, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, and 16 others. 2024. FOLIO: Natural language reasoning with first-order logic. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22017–22031, Miami, Florida, USA. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Ryan Hsiang, William Adkisson, Robert Joseph George, and Anima Anandkumar. 2025. Leandojo-v2: A comprehensive library for AI-assisted theorem proving in lean. In *The 5th Workshop on Mathematical Reasoning and AI at NeurIPS 2025*.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Shashank Kirtania, Priyanshu Gupta, and Arjun Radhakrishna. 2024. LOGIC-LM++: Multi-step refinement for symbolic formulations. In *Proceedings of the 2nd Workshop on Natural Language Reasoning and Structured Explanations (@ACL 2024)*, pages 56–63, Bangkok, Thailand. Association for Computational Linguistics.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, page 611–626, New York, NY, USA. Association for Computing Machinery.

Oleksii Levkovskyi and Wei Li. 2021. Generating predicate logic expressions from natural language. In *SoutheastCon 2021*, pages 1–8.

Francisco F. López-Ponce and Gemma Bel-Enguix. 2025. Into the limits of logic: Alignment methods for formal logical reasoning. In *Proceedings of The 3rd Workshop on Mathematical Natural Language Processing (MathNLP 2025)*, pages 112–123, Suzhou, China. Association for Computational Linguistics.

Haoran Lu, Luyang Fang, Ruidong Zhang, Xinliang Li, Jiazhang Cai, Huimin Cheng, Lin Tang, Ziyu Liu, Zeliang Sun, Tao Wang, Yingchuan Zhang, Arif Hassan Zidan, Jinwen Xu, Jincheng Yu, Meizhi Yu, Hanqi Jiang, Xilin Gong, Weidi Luo, Bolun Sun, and 31 others. 2025. Alignment and safety in large language models: Safety mechanisms, training paradigms, and emerging challenges. *Preprint*, arXiv:2507.19672.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, Benjamin Bossan, and Marian Tietz. 2022. PEFT: State-of-the-art parameter-efficient fine-tuning methods. https://github.com/huggingface/peft.

W. McCune. 2005–2010. Prover9 and mace4. http://www.cs.unm.edu/~mccune/prover9/.

Theo Olausson, Alex Gu, Ben Lipkin, Cedegao Zhang, Armando Solar-Lezama, Joshua Tenenbaum, and Roger Levy. 2023. LINC: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5153–5176, Singapore. Association for Computational Linguistics.

OpenAI. 2025. gpt-oss-120b & gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

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.

Xin Quan, Marco Valentino, Louise A. Dennis, and Andre Freitas. 2025. Faithful and robust LLM-driven theorem proving for NLI explanations. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 17734–17755, Vienna, Austria. Association for Computational Linguistics.

Qwen Team. 2026. Qwen3.5: Towards native multimodal agents.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: your language model is secretly a reward model. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Leonardo Ranaldi, Marco Valentino, and Andre Freitas. 2025. Improving chain-of-thought reasoning via quasi-symbolic abstractions. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 17222–17240, Vienna, Austria. Association for Computational Linguistics.

Abulhair Saparov and He He. 2023. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *The Eleventh International Conference on Learning Representations*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Ziju Shen, Naohao Huang, Fanyi Yang, Yutong Wang, Guoxiong Gao, Tianyi Xu, Jiedong Jiang, Wanyi He, Pu Yang, Mengzhou Sun, Haocheng Ju, Peihao Wu, Bryan Dai, and Bin Dong. 2025. Real-prover: Retrieval augmented lean prover for mathematical reasoning. *Preprint*, arXiv:2505.20613.

Christian Szegedy. 2020. A promising path towards autoformalization and general artificial intelligence. In *Intelligent Computer Mathematics*, pages 3–20, Cham. Springer International Publishing.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2021. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634, Online. Association for Computational Linguistics.

Zhi Rui Tam, Cheng-Kuang Wu, Yi-Lin Tsai, Chieh-Yen Lin, Hung-yi Lee, and Yun-Nung Chen. 2024. Let me speak freely? a study on the impact of format restrictions on large language model performance. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1218–1236, Miami, Florida, US. Association for Computational Linguistics.

Gemma Team. 2025a. Gemma 3.

Qwen Team. 2025b. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. 2020. TRL: Transformers Reinforcement Learning.

Qingxiang Wang, Cezary Kaliszyk, and Josef Urban. 2018. First experiments with neural translation of informal to formal mathematics. *Preprint*, arXiv:1805.06502.

Ke Weng, Lun Du, Sirui Li, Wangyue Lu, Haozhe Sun, Hengyu Liu, and Tiancheng Zhang. 2025. Autoformalization in the era of large language models: A survey. *Preprint*, arXiv:2505.23486.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Huggingface's transformers: State-of-the-art natural language processing. *Preprint*, arXiv:1910.03771.

Yuhuai Wu, Albert Qiaochu Jiang, Wenda Li, Markus Rabe, Charles Staats, Mateja Jamnik, and Christian Szegedy. 2022. Autoformalization with large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 32353–32368. Curran Associates, Inc.

Lei Xu, Pierre Beckmann, Marco Valentino, and Andre Freitas. 2026. Adaptive LLM-symbolic reasoning via dynamic logical solver composition. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1187–1208, Rabat, Morocco. Association for Computational Linguistics.

Kaiyu Yang, Aidan M. Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan Prenger, and Anima Anandkumar. 2023. Leandojo: theorem proving with retrieval-augmented language models. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Yuan Yang, Siheng Xiong, Ali Payani, Ehsan Shareghi, and Faramarz Fekri. 2024. Harnessing the power of large language models for natural language to first-order logic translation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6942–6959, Bangkok, Thailand. Association for Computational Linguistics.

Lan Zhang, Xin Quan, and Andre Freitas. 2024. Consistent autoformalization for constructing mathematical libraries. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4020–4033, Miami, Florida, USA. Association for Computational Linguistics.

Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. 2022. minif2f: a cross-system benchmark for formal olympiad-level mathematics. In *International Conference on Learning Representations*.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2020. Fine-tuning language models from human preferences. *Preprint*, arXiv:1909.08593.

A Prompts

A.1 Translation

GIVEN A SET OF PREMISES, THE TASK IS TO PARSE THE PROBLEM AND THE QUESTION INTO FIRST-ORDER LOGIC FORMULARS. ANSWER ONLY WITH THE TRANSLATED PREMISES. THE GRAMMAR OF THE FIRST-ORDER LOGIC FORMULAR IS DEFINED AS FOLLOWS:

1 LOGICAL CONJUNCTION OF EXPR1 AND EXPR2: $expr1 \wedge expr2$

2 LOGICAL DISJUNCTION OF EXPR1 AND EXPR2: $expr1 \vee expr2$

3 LOGICAL EXCLUSIVE DISJUNCTION OF EXPR1 AND EXPR2: $expr1 \oplus expr2$

4 LOGICAL NEGATION OF EXPR1: $\neg expr1$

5 EXPR1 IMPLIES EXPR2: $expr1 \rightarrow expr2$

6 EXPR1 IF AND ONLY IF EXPR2: $expr1 \leftrightarrow expr2$

7 LOGICAL UNIVERSAL QUANTIFICATION: $\forall x$

8 LOGICAL EXISTENTIAL QUANTIFICATION: $\exists x$

NATURAL LANGUAGE PREMISES:

- ALL PEOPLE WHO REGULARLY DRINK COFFEE ARE DEPENDENT ON CAFFEINE.
- PEOPLE EITHER REGULARLY DRINK COFFEE OR JOKE ABOUT BEING ADDICTED TO CAFFEINE.
- NO ONE WHO JOKES ABOUT BEING ADDICTED TO CAFFEINE IS UNAWARE THAT CAFFEINE IS A DRUG.
- RINA IS EITHER A STUDENT AND UNAWARE THAT CAFFEINE IS A DRUG, OR NEITHER A STUDENT NOR UNAWARE THAT CAFFEINE IS A DRUG.
- IF RINA IS NOT A PERSON DEPENDENT ON CAFFEINE AND A STUDENT, THEN RINA IS EITHER A PERSON DEPENDENT ON CAFFEINE AND A STUDENT, OR NEITHER A PERSON DEPENDENT ON CAFFEINE NOR A STUDENT.

PREDICATES:

- $Dependent(x)$ ::: X IS A PERSON DEPENDENT ON CAFFEINE.
- $Drinks(x)$ ::: X REGULARLY DRINKS COFFEE.
- $Jokes(x)$ ::: X JOKES ABOUT BEING ADDICTED TO CAFFEINE.
- $Unaware(x)$ ::: X IS UNAWARE THAT CAFFEINE IS A DRUG.
- $Student(x)$ ::: X IS A STUDENT.

FOL PREMISES:

- $\forall x(Drinks(x) \rightarrow Dependent(x))$ ::: ALL PEOPLE WHO REGULARLY DRINK COFFEE ARE DEPENDENT ON CAFFEINE.
- $\forall x(Drinks(x) \oplus Jokes(x))$ ::: PEOPLE EITHER REGULARLY DRINK COFFEE OR JOKE ABOUT BEING ADDICTED TO CAFFEINE.
- $\forall x(Jokes(x) \rightarrow Unaware(x))$ ::: NO ONE WHO JOKES ABOUT BEING ADDICTED TO CAFFEINE IS UNAWARE THAT CAFFEINE IS A DRUG.
- $(Student(rina) \wedge Unaware(rina)) \oplus \neg(Student(rina) \vee Unaware(rina))$ ::: RINA IS EITHER A STUDENT AND UNAWARE THAT CAFFEINE IS A DRUG, OR NEITHER A STUDENT NOR UNAWARE THAT CAFFEINE IS A DRUG.
- $\neg(Dependent(rina) \wedge Student(rina)) \rightarrow (Dependent(rina) \wedge Student(rina)) \oplus \neg(Dependent(rina) \vee Student(rina))$ ::: IF RINA IS NOT A PERSON DEPENDENT ON CAFFEINE AND A STUDENT, THEN RINA IS EITHER A PERSON DEPENDENT ON CAFFEINE AND A STUDENT, OR NEITHER A PERSON DEPENDENT ON CAFFEINE NOR A STUDENT.

NATURAL LANGUAGE PREMISES:
{} # Brackets for f-strings.

FOL PREMISES:

A.2 Inference

GIVEN A SET OF PREMISES AND CONCLUSION IN FIRST-ORDER LOGIC, YOUR TASK IS TO DETERMINE THE LOGICAL VALIDITY OF THE CONCLUSION: TRUE, FALSE, OR UNCERTAIN. ANSWER ONLY WITH THE LOGICAL VALUE. A TRUE CONCLUSION IS ONE THAT CAN BE OBTAINED VIA A VALID INFERENCE PROCEDURE FROM THE GIVEN PREMISES. A FALSE CONCLUSION IS ONE THAT CONTRADICTS ONE OR MORE PREMISES DURING THE INFERENCE PROCEDURE. AN UNCERTAIN CONCLUSION IS NEITHER TRUE NOR FALSE. MEANING THAT THERE IS INSUFFICIENT INFORMATION IN THE PREMISES TO INFER IT, BUT THE CONCLUSION IT SELF DOESN'T CONTRADICT ANY PREMISE. THE FOLLOWING EXAMPLE SHOWS A SET OF PREMISES AND CONCLUSIONS WHERE EACH CONCLUSION REPRESENTS A DIFFERENT LOGICAL VALIDITY. YOU SHOULD ANSWER SIMILARLY.
FOL-PREMISES:

- $\forall x(WorkAt(x, meta) \rightarrow HighIncome(x))$
- $\forall x(HighIncome(x) \rightarrow \neg MeansToDestination(x, bus))$
- $\forall x(MeansToDestination(x, bus) \oplus MeansToDestination(x, drive))$
- $\forall x(HaveCar(x) \rightarrow MeansToDestination(x, drive))$
- $\forall x(Student(x) \rightarrow \neg MeansToDestination(x, drive))$
- $HaveCar(james) \vee WorkAt(james, meta)$

FOL-CONCLUSION:

- $MeansToDestination(x, drive) \vee Student(james)$
- $Student(james)$
- $\neg HighIncome(james)$

ANALYSIS: THE FIRST CONCLUSION IS TRUE. PREMISE 6 STATES THAT EITHER JAMES HAS A CAR (IN WHICH CASE PREMISE 4 GIVES US THE CONCLUSION) OR JAMES WORKS AT META (IN WHICH CASE PREMISE 4 IMPLIES PREMISE 2, WHICH COMBINED WITH PREMISE 3 GIVES US THE CONCLUSION). THE SECOND CONCLUSION IS FALSE. PREMISE 5 STATES THAT STUDENTS CAN'T HAVE A CAR AS A MEANSTODESTINATION, HOWEVER THE FIRST CONDITION TELLS US JAMES HAS SUCH MEANS. THE THIRD CONDITION IS UNCERTAIN. PREMISE 1 IS THE ONLY GUARANTEE TO HAVE A HIGH INCOME, HOWEVER WE CAN'T DETERMINE THAT JAMES WORKS AT META (PREMISE 6).

FOL-PREMISES:
{} # Brackets for f-strings.

FOL-CONCLUSION:
{} # Brackets for f-strings.

ANSWER:

A.3 Back-translation

GIVEN A SINGLE PREMISE IN FIRST-ORDER LOGIC, YOUR TASK IS TO TRANSLATE THE PREMISE INTO NATURAL LANGUAGE. ANSWER ONLY WITH THE TRANSLATED PREMISE. IT SHOULD BE A SINGLE SENTENCE. THE GRAMMAR OF THE FIRST-ORDER LOGIC FORMULAR IS DEFINED AS FOLLOWS:

1. LOGICAL CONJUNCTION OF EXPR1 AND EXPR2: $expr1 \wedge expr2$
2. LOGICAL DISJUNCTION OF EXPR1 AND EXPR2: $expr1 \vee expr2$
3. LOGICAL EXCLUSIVE DISJUNCTION OF EXPR1 AND EXPR2: $expr1 \oplus expr2$
4. LOGICAL NEGATION OF EXPR1: $\neg expr1$

5 EXPR1 IMPLIES EXPR2: $expr1 \rightarrow expr2$

6 EXPR1 IF AND ONLY IF EXPR2: $expr1 \leftrightarrow expr2$

7 LOGICAL UNIVERSAL QUANTIFICATION: $\forall x$

8 LOGICAL EXISTENTIAL QUANTIFICATION: $\exists x$

BELOW ARE EXAMPLES OF THE TRANSLATION:
PREMISES:

- $\neg(PartTime(jackie) \oplus ForbesList(jackie)) \rightarrow \exists y(LessThan(y, num2) \wedge TakesCourses(x, y)) \wedge ForbesList(jackie)$
- $\neg In(borjMasouda, tunisia)$

NATURAL LANGUAGE:

- IF JACKIE EITHER ENROLLS AS PART-TIME IN THE CURRENT SEMESTER AND IS LISTED IN THE FORBES 30 UNDER 30, OR NEITHER ENROLLS AS PART-TIME IN THE CURRENT SEMESTER NOR IS LISTED IN THE FORBES 30 UNDER 30, THEN JACKIE TAKES LESS THAN TWO COURSES IN THE CURRENT SEMESTER AND LISTED IN THE FORBES 30 UNDER 30.
- BORJ MASOUDA IS NOT IN TUNISIA.

———

PREMISE:
{}
NATURAL LANGUAGE:

B Generation Hyperparameters

Answer generation throughout the pipeline was carried out in an NVIDIA RTX 5000 Ada Generation. Baseline and syntax restricted results where generated using vLLM (Kwon et al., 2023), whereas alignment results where generated using the Transformers library (Wolf et al., 2020).

Tables 2 and 3 show the hyperparameters of the instances of the vLLM classes SamplingParams and LLM. Unmentioned parameters were kept as default. Generation time varied anywhere from 6 to 100 minutes per model (GPT-20B being the fastest, Q3-8B the slowest). This is a considerable improvement over HuggingFace's base implementation, where a single step would take a minimum of 5 hours regardless of the model.

KTO models where evaluated using HuggingFace's Transformers directly due to unresolved version discrepancies between TRL 1.0 and vLLM. Table 4 shows the GenerationConfig hyperparameters.

Hyperparameter	Value
TEMPERATURE	0.2
TOP_P	0.9
MAX_TOKENS	5000
SEED	67
LOGPROBS	1

Table 2: SamplingParams instance hyperparameters. Logprobs doesn't affect generation, it merely returns the logits needed to calculate perplexity.

Hyperparameter	Value
MAX_MODEL_LEN	7000
QUANTIZATION	'fp8'
ENFORCE_EAGER	True
LIMIT_MM_PER_PROMPT	{'image': 0, 'video': 0}

Table 3: LLM instance hyperparameters. FP8 quantization was used for every model except GPT-OSS-20B which used mxfp4, and the DeepSeek distilled models which were deployed without quantization.

C Manual Filtering and Model Discussion

C.1 Filtering

The prompt used for these experiments has a defined structure of: PROBLEM DEFINITION —— (Dashes) EXAMPLE BEHAVIOR —— (Dashes) INSTANCE TO SOLVE. This makes filtering easy by splitting the response using dashes. However, there is no golden rule for automatic filtering since models often use different individual characters and semantic structures for splitting and formatting their answer. Filtering was done using a common filtering function complemented by manual revision of outputs.

C.1.1 Autoformalization

Responses often contain the translated premises followed by hallucinated premises and their corresponding translation. Such premises are unrelated with the current instance, at best they correspond to the example behavior used in the prompt. Loops in autoformalization occur with hallucinated premises and premises with long constant names (often related to uncommon proper nouns).

C.1.2 Inference and Back-Translation

Filtering is simpler for steps 2 and 3 since the answers themselves are shorter and often presented in a single sentence. However, generation loops do occur with thinking models. Loops in this case correspond to inconclusive internal reasoning debates,

Hyperparameter	Value
TEMPERATURE	0.2
MAX_NEW_TOKENS	1000
OUTPUT_LOGITS	True
DO_SAMPLE	True
RETURN_FULL_TEXT	False

Table 4: GenerationConfig. Other parameters were kept as default.

regardless of the step.

C.2 Model Discussion

Models endowed with a Thinking/Reasoning mode (all but the Gemma 3 family) use it. This means models have an internal analysis prior to their final output. The internal debate often contaminates the answer's format, and in some cases models fall in a loop during the debate until exhausting generation tokens without presenting a final answer.

C.2.1 DeepSeek-R1 Distilled

Four models were originally tested, two (DEEPSEEK-R1-DISTILL-LLAMA-8B and DEEPSEEK-R1-DISTILL-QWEN-1.5B) were dropped and not reported due to a highly unstable set of generations. Automatic and manual filtering for such checkpoints yielded unusable results.

The two reported models (DEEPSEEK-R1-0528-QWEN3-8B and DEEPSEEK-R1-DISTILL-QWEN-7B) presented inconsistencies with response format, often repeated strings until token limit, and ran out of thinking tokens before outputting a response (even if they had reached a correct answer during the thinking process). DEEPSEEK-R1-DISTILL-QWEN-7B was the only model that had to be fully manually processed.

C.2.2 Gemma 3 and Qwen3

These families of models answered directly and with no preamble, while following generation structure on each step for all models. Automatic filter worked with all models and only needed manual adjustment in less than 10% of total cases.

C.2.3 GPT-Oss

A third of the samples were manually filtered due to inconsistent generation patterns. Manual filtering had to be carried out since this model didn't follow the use of dashes to separate parts of the generated answer.

D LogicSim+

Steps 1 and 3 are evaluated in a different level of granularity due to inherent differences between FOL and NL, even if conceptually we're evaluating pairs of premises. $LogicSim+_t$ (Step 1, FOL vs FOL) manages to extract differences in predicates, premises, or logical operators in an isolated manner. This is due to FOL syntax restrictions and how they identify what can be or can't be a particular element without the need of context. Logical operators are static symbols, variables/constants can never be followed by an open parenthesis, predicates always have a variable/constant associated to them within parenthesis (these aren't the actual definitions of each value, however such patterns can be used for automatic extraction of the actual values).

$LogicSim+_b$ (Step 3, FOL vs NL) is unable to extract information in a similar manner since there's no unique correlation between elements in FOL and NL, e.g. the FOL Predicate $NationalPark(x)$ and the constant $publicMarketplace$ are expressed in NL using a single noun (park/marketplace) and adjective (national/public), however they correspond to two different logically valid elements. The inherent flexibility of syntactic structure in NL forces us to simplify measurements in order to obtain a degree of similarity.

D.1 Back-Translation

Consider the following premise expressed in FOL:

Athlete(stephenCurry) ∧ Professional(stephenCurry) ∧ CenterBack(stephenCurry)

The following three sentences show variations of translations that exemplify compressed, explicit, and extended translations.

A **compressed** translation is: `Stephen Curry is an athlete and a professional and a center-back.` (generated by DEEPSEEK-R1-0528-QWEN3-8B). Stephen Curry is mentioned only once, meaning that less information is being expressed explicitly.

An **explicit** translation is: `Stephen Curry is an athlete, and Stephen Curry is a professional, and Stephen Curry is a center-back.` (Human-generated, based on the QWEN3.5-4B back translation). This translation states each individual premise and logical operator.

An **extended** translation is: `Stephen Curry is an athlete, and Stephen Curry is a`

`professional, and it is also the case that Stephen Curry plays as a centerback. Stephen Curry is all of those things.` This translation doesn't alter the logical value of the FOL premise, but mentions Stephen Curry an extra time.

E Alignment

E.1 Training Details and Parameters

Alignment was done out using TRL (von Werra et al., 2020). Post-training was carried out in a single NVIDIA RTX 5000 Ada Generation.

E.1.1 KTO

Trained models were quantized using BITSANDBYTES (4-bit quantization) (Dettmers et al., 2021; Dettmers and Zettlemoyer, 2023) and preprocessed using PEFT (Mangrulkar et al., 2022) and LoRA (Hu et al., 2022). Training was done using TRL. Tables 5 and 6 show full hyperparameters.

Hyperparameter	Value
R	16
LORA_ALPHA	8
TARGET_MODULES	"all-linear"
LORA_DROPOUT	0.05
BIAS	"none"
TASK_TYPE	"CAUSAL_LM"

Table 5: LoRA Configuration. Equal for all models.

Hyperparameter	Value
LEARNING_RATE	5e-6
NUM_TRAIN_EPOCHS	2
PER_DEVICE_TRAIN_BATCH_SIZE	2, 4, 5
USE_CACHE	False

Table 6: KTO Configuration. Batch size is determined based on highest value that doesn't lead to an Out-Of-Memory error. The values shown correspond to Gemma3, Qwen3, DeepSeek accordingly.

E.2 Alignment Datasets

General purpose alignment datasets cover a broad set of topics and responses, and tend to be orders of magnitude bigger than our KTO datasets (60K + samples). Our datasets focus solely on our three step pipeline, but can be easily integrated into a bigger preference dataset or be slightly modified to accommodate different training algorithms. These experimental variations can set grounds for future work.

E.3 FOLIO Cleaning details

FOL premises were fixed in a case by case basis following a conservative policy: only in cases where modifying the original text's preserves the logical value do we actual modify it. This leads to preserving author intent over re-derivation, and is validated by running the grammar over the modified FOL premises.

Error Class	Frequency
EXTRA OPENING PAREN	56.5% (165)
EXTRA CLOSING PAREN	39.0% (114)
QUANTIFIER WITHOUT VARIABLE	1.7% (5)
MISGROUPED PARENS	1.4% (4)
STRAY ANNOTATION	1.0% (3)
MISSING ARGUMENT COMMA	0.3% (1)
TOTAL	100.0% (292)

Table 7: Parse failures in the original FOLIO release, by error class.

Edit Distance	Frequency
1	79.9% (235)
2	9.9% (29)
3	8.2% (24)
4	0.3% (1)
5	0.7% (2)
29	1.0% (3)
TOTAL	100.0% (294)

Table 8: Levenshtein distance between each changed premise in the original FOLIO release and its corrected counterpart. Unchanged premises (8,717 of 9,011) are omitted. The single distance-29 cluster corresponds to the trailing English annotation removed in three occurrences.

F Context Free Grammars

Evaluation was carried out over answers created under the Generation Grammar. However, FOLIO parsing and revision was done using Evaluation Grammar.

Generation Grammar

$$
\begin{aligned}
\varphi &::= \varphi \leftrightarrow \varphi \mid \varphi \rightarrow \varphi \mid \varphi \vee \varphi \mid \varphi \oplus \varphi \mid \varphi \wedge \varphi \\
&\mid \forall \bar{x}\, \varphi \mid \exists \bar{x}\, \varphi \mid \neg\varphi \\
&\mid P(t_1, \ldots, t_n) \mid t = t \mid \neg(t = t) \mid (\varphi) \\
t &::= x \mid c \mid n \mid f(t_1, \ldots, t_n) \\
\bar{x} &::= x \mid x\, \bar{x}
\end{aligned}
$$

Evaluation Grammar

$$
\begin{aligned}
\psi \ &::= \psi\,(\leftrightarrow|\Longleftrightarrow)\,\psi \mid \psi\,(\rightarrow|\longrightarrow)\,\psi \\
&\mid \psi \vee \psi \mid \psi \oplus \psi \mid \psi \wedge \psi \mid \neg\psi \\
&\mid \forall\,\bar{x}\,\psi \mid \exists\,\bar{x}\,\psi \\
&\mid P(t_1, \ldots, t_n) \mid t = t \mid t \neq t \\
&\mid \neg(t = t) \mid (\psi) \\
t \ &::= x \mid c \mid n \mid f(t_1, \ldots, t_n) \\
\bar{x} \ &::= x \mid x\,\bar{x}
\end{aligned}
$$

G Results

Table 9 shows extra information regarding results and incorrect generations. Step 2 is not evaluated for syntax restrained generation, hence column one is left empty for such strategy.

Strategy	Model	Bad Inference	Arity	Syntax	N/A Trans
One-Shot	Q3-4B	**(0, 0)**	(17, 32)	(37, 34)	(0, 0)
	Q3-8B	(1, 5)	(22, 22)	(**29**, 38)	(8, 3)
	Q3-14B	**(0, 0)**	(13, 15)	(45, 46)	(1, 0)
	Q3.5-4B	(0, 1)	(16, 13)	(44, 55)	(3, 3)
	Q3.5-9B	**(0, 0)**	(12, 20)	(53, 61)	(0, 2)
	G3-4B	**(0, 0)**	(27, 36)	(78, 60)	(0, 0)
	G3-12B	(4, 8)	(17, 25)	(39, 44)	(0, 0)
	GPT-20B	(2, 2)	**(2, 3)**	(50, **24**)	(0, 0)
	DS-0528	(8, 3)	(14, 8)	(63, 61)	(0, 0)
	DS-DIST	(27, 34)	(17, 20)	(51, 41)	(13, 0)
Syntax	Q3-4B	-	(13, 34)	(66, 75)	(0, 0)
	Q3-8B	-	(19, 26)	(61, 71)	(0, 0)
	Q3-14B	-	(25, 12)	(101, 121)	(0, 0)
	Q3.5-4B	-	(13, 18)	(64, 88)	(0, 0)
	Q3.5-9B	-	(23, 26)	(58, 76)	(0, 0)
	G3-4B	-	(22, 25)	(53, 62)	(0, 0)
	G3-12B	-	(28, 26)	(30, 37)	(0, 0)
	GPT-20B	-	(**7**, **3**)	(33, 22)	(0, 0)
	DS-0528	-	(18, 30)	(36, 30)	(0, 0)
	DS-DIST	-	(14, 17)	(42, 46)	(0, 0)
KTO	Q3-14B	**(0, 0)**	(**18, 19**)	(27, **27**)	(0, 0)
	G3-12B	**(0, 0)**	(21, 32)	(**26**, 47)	(0, 0)
	DS-0528	**(0, 0)**	(26, 25)	(55, 45)	(0, 0)

Table 9: Additional information regarding reported results. **Bad Inference** corresponds to the amount of tags that aren't assigned to a truth-value in step 2. **Arity** and **Syntax** correspond to the amount of Prover9 related errors. All values are once again reported in the (Validation, Test) format.