Evaluating LLMs Capabilities at Natural Language to Logic Translation: A Preliminary Investigation

Andrea Brunello[1], Riccardo Ferrarese[1], Luca Geatti[1], Enrico Marzano[2], Angelo Montanari[1] and Nicola Saccomanno[1,*]

[1]University of Udine, Italy
[2]G.A.P. Srl, Italy

Abstract

Translating natural language (NL) into logical formalisms like First-Order Logic (FOL) has long been a challenge across multiple disciplines, including mathematics, computer science, and education. Traditional computational linguistics methods have struggled with this task due to the complexity and ambiguity of natural language. However, advancements in Natural Language Processing (NLP), particularly the introduction of Large Language Models (LLMs), have opened up new possibilities for tackling this challenge. Despite their potential, a systematic approach to evaluating the performance of LLMs in NL-to-FOL translation is still lacking. In this study, we take a first step towards filling in this gap. We examine a large dataset based on students' efforts in formalizing natural language statements from the book *"Language, Proof, and Logic"*. Based on this dataset, we propose a preliminary evaluation pipeline to assess LLM performance in NL-to-FOL translation tasks, considering both syntactic and semantic aspects. We then apply this pipeline to evaluate two recent LLMs, Meta's Llama 3.1 (8B) and Google DeepMind's Gemma 2 (9B). Our findings validate the proposed approach, revealing key similarities and differences between LLM-generated and student-produced formulas, and provide valuable insights into the current capabilities of LLMs in this domain.

Keywords

Natural Language Processing, Large Language Models, Formal Methods, First Order Logic, Translation

1. Introduction

Translating natural language (NL) into logical formalisms like First-Order Logic (FOL) is a longstanding challenge that spans multiple disciplines, including mathematics, computer science, and education. In mathematics, this translation facilitates automated theorem proving [1], in computer science, it helps in developing systems that approximate human reasoning [2], in engineering, it supports domain experts with the definition of formal requirements that should be verified at runtime by a systems [3], and in education, it gives a precious support to analyze common mistakes made by students [4].

Historically, traditional computational linguistics methods have attempted to tackle this task, but with limited success, often struggling to maintain accuracy and consistency when dealing with complex or ambiguous sentences (see, for instance, the survey paper [5]). For example, in the domain of coreference resolution—the task of determining when different expressions in a text refer to the same entity—traditional methods frequently fall short in cases involving pronoun ambiguity or nested references. However, recent advancements in natural language processing (NLP), particularly with the advent of large language models (LLMs), offer new possibilities. Models such as GPT and BERT have demonstrated an exceptional capacity in processing and generating natural language texts, excelling in tasks like translation, summarization, sentiment analysis, question answering, and even code generation [6]. Their ability to handle context and perform sophisticated reasoning makes them promising candidates also for translating natural language statements into logic formulas. Indeed, several LLM-based

OVERLAY 24: 6th International Workshop on Artificial Intelligence and fOrmal VERification, Logic, Automata, and sYnthesis
*Corresponding author.
andrea.brunello@uniud.it (A. Brunello); ferrarese.riccardo@spes.uniud.it (R. Ferrarese); luca.geatti@uniud.it (L. Geatti); e.marzano@gapitalia.it (E. Marzano); angelo.montanari@uniud.it (A. Montanari); nicola.saccomanno@uniud.it (N. Saccomanno)
0000-0003-2063-218X (A. Brunello); 0000-0002-7125-787X (L. Geatti); 0009-0001-1469-6497 (E. Marzano); 0000-0002-4322-769X (A. Montanari); 0000-0001-5916-3195 (N. Saccomanno)

Table 1
Exemplary instances from the dataset

Student_id	Sentence	Answer	N_attempt	Correct
1	'a' is a cube	$Cube(a)$	1	True
1	'b' is smaller than 'a'	$Smaller(b, a)$	1	True
2	'b' is smaller than 'a'	$Larger(b, a)$	1	False
2	'b' is smaller than 'a'	$Larger(a, b)$	2	True
...	...	...	...	...
1756	Nothing is between objects of shapes other than its own	`***omitted as requested by the dataset authors***`	4	True

methods have recently been proposed for this task [7, 8, 9, 10]. The diversity of these methods and the datasets used highlights the need for standardized benchmarks to systematically and comparably assess the effectiveness of the models. However, to the best of our knowledge, no such systematic approach has yet been proposed.

Our study represents a first step in this direction. We examine a large dataset derived from research on the challenges students face when formalizing natural language statements into FOL [11, 12, 13, 14], as outlined in the book *"Language, Proof, and Logic"* [15]. Based on this dataset, we propose an initial pipeline to evaluate the performance of LLMs in translating natural language phrases into first-order logic formulas, considering both syntactic and semantic aspects. We then apply this pipeline to assess the performance of the two recent LLMs Meta's LLM Llama 3.1 (8B) [16, 17] and Google DeepMind's Gemma 2 (9B) [18]. While our work is not yet intended as a full-fledged benchmark, the experimental results validate the proposed approach and highlight key similarities and differences between LLM-generated formulas and those produced by students, offering valuable insights into the current capabilities of LLMs in this domain. These findings warrant further exploration and, in addition, emphasize the need for continued investigation into the interaction between the fields of NLP and formal methods.

The paper is structured as follows: Section 2 introduces the dataset and preprocessing steps undertaken for analysis. Section 3 outlines our evaluation pipeline for assessing LLM performance on NL-to-FOL translation tasks, which is then experimentally evaluated in Section 4. Finally, Section 5 provides concluding remarks, discusses the limitations of the current study and outlines future research directions.

2. Dataset

The dataset we considered is an extract of the years 2001–2010 from the *Grade Grinder Corpus Release 1.0* [11], kindly provided to us privately by the authors. It consists of correct and incorrect answers, continuously collected through a dedicated tutoring software, from students responding to questions posed in the book *"Language, Proof, and Logic"* [15]. Each question involves a natural language utterance describing a situation in Tarski's World [19], where students are tasked with formulating a corresponding First-Order Logic (FOL) expression to encode the given text.

Our extract contains 16,265,166 answers, both correct and incorrect, submitted by 50,608 students. After a pre-processing phase, each answer includes a student identifier, the logic formula written by the student, the corresponding natural language phrase, and a Boolean flag indicating whether the answer is correct. Additionally, each entry contains an attribute indicating the sequential attempt number, as students may make multiple attempts to answer the same question while interacting with the tutoring software (see Table 1). Note that there may be more than one correct answer per phrase due to the equivalence of different logic formulas and the flexibility of the allowed syntax, which includes relations along with their opposites. This is summarized in Table 2.

Overall, answers refer to 208 distinct natural language phrases, which we categorized along 6 difficulty levels. To such an extent, based on the work of [13], for each phrase we calculated the percentage of failed first-attempt answers as well as the so-called *stickiness* value, i.e., the average number of attempts made by students before reaching a correct answer. We then performed K-means clustering on these

Table 2
Syntax allowed for the formulas

Relation/arity	Tet/1, Cube/1, Dodec/1, Small/1, Medium/1, Large/1, SameShape/2, SameSize/2, Larger/2, Smaller/2, SameCol/2, SameRow/2, Adjoins/2, FrontOf/2, BackOf/2, RightOf/2, LeftOf/2, =/2, Between/3
Connectives	$\neg, \wedge, \vee, \rightarrow, \leftrightarrow$
Quantifiers	$\exists, \forall$
Constants	a, b, c, d, e, f
Variables	s, t, u, v, w, x, y, z
Structure	(,)

Table 3
Difficulty levels

Difficulty level	N. of phrases	N. of answers	% Failed first attempts	Average stickiness
super-low	94	8,546,754	5.10	1.11
low	62	4,578,148	13.15	1.30
mid-low	13	810,726	23.53	1.58
mid-high	29	1,857,054	37.89	2.06
high	8	378,221	42.35	2.48
super-high	2	94,263	52.65	3.09

two variables, identifying 6 clusters relying on an instance separation criterion; they are summarized in Table 3. Note how, overall, the number of phrases decreases as the difficulty level increases. For each phrase, the ground truth is given by the set of (unique) correctly submitted formulas, a number which ranges from a minimum of 1 to a maximum of 1150, with an average of 99.

3. NL-to-FOL Evaluation Pipeline

In this section, we provide an account of the pipeline we developed to evaluate natural language to formula translation, which is summarized in Figure 1. Suppose an LLM is given a natural language phrase p, belonging to our considered dataset, that describes a situation in Tarski's World; in response, it generates a formal representation of this phrase as a FOL formula ϕ. To evaluate the syntactic and semantic correctness of ϕ, we proceed as follows:

1. The formula ϕ is parsed using Python's `Lark` library [20], following a specifically designed FOL grammar describing formulas of Tarski's World (see Table 2). The parsing may fail, due to grammar violations. In such case, we rely on a set of hard-coded rules, primarily based on regular expressions, to identify these kinds of syntax errors:
   - *Relation/arity:* a relation that is not part of the defined syntax or is used with incorrect arity;
   - *Connectives and quantifiers:* non-adherence to the syntax or misuse of a connective (e.g., $\wedge$ connective without one of the conjuncts);
   - *Constants and variables:* non-adherence to the naming conventions;
   - *Structure:* unpaired or misused parentheses;
   - *Other:* residual category for the parsing errors which could not be automatically classified.

   At this point, the errors identified in the formula are kept track of, the formula is labeled as both syntactically and semantically incorrect, and the pipeline ends. Otherwise, if the parsing of ϕ succeeds, the formula is passed down to the next point in the pipeline;
2. Relying on Python's ZSS library [21], we calculate the *tree edit distance* [22] between ϕ and each possible ground truth for the phrase p, identifying the most similar formula ψ;
3. We evaluate the logical equivalence between ϕ and ψ relying on the `Z3` theorem prover [23]. Specifically, we verify the unsatisfiability of $\neg(\phi \leftrightarrow \psi)$;
4. If the two formulas are found to be equivalent, ϕ is labeled as both syntactically and semantically correct. Otherwise, if they are non-equivalent, the formula is labeled as syntactically correct but not semantically correct, and we also output the distance calculated at Point 2., with the idea that it represents the cost of semantically transforming ϕ into ψ.

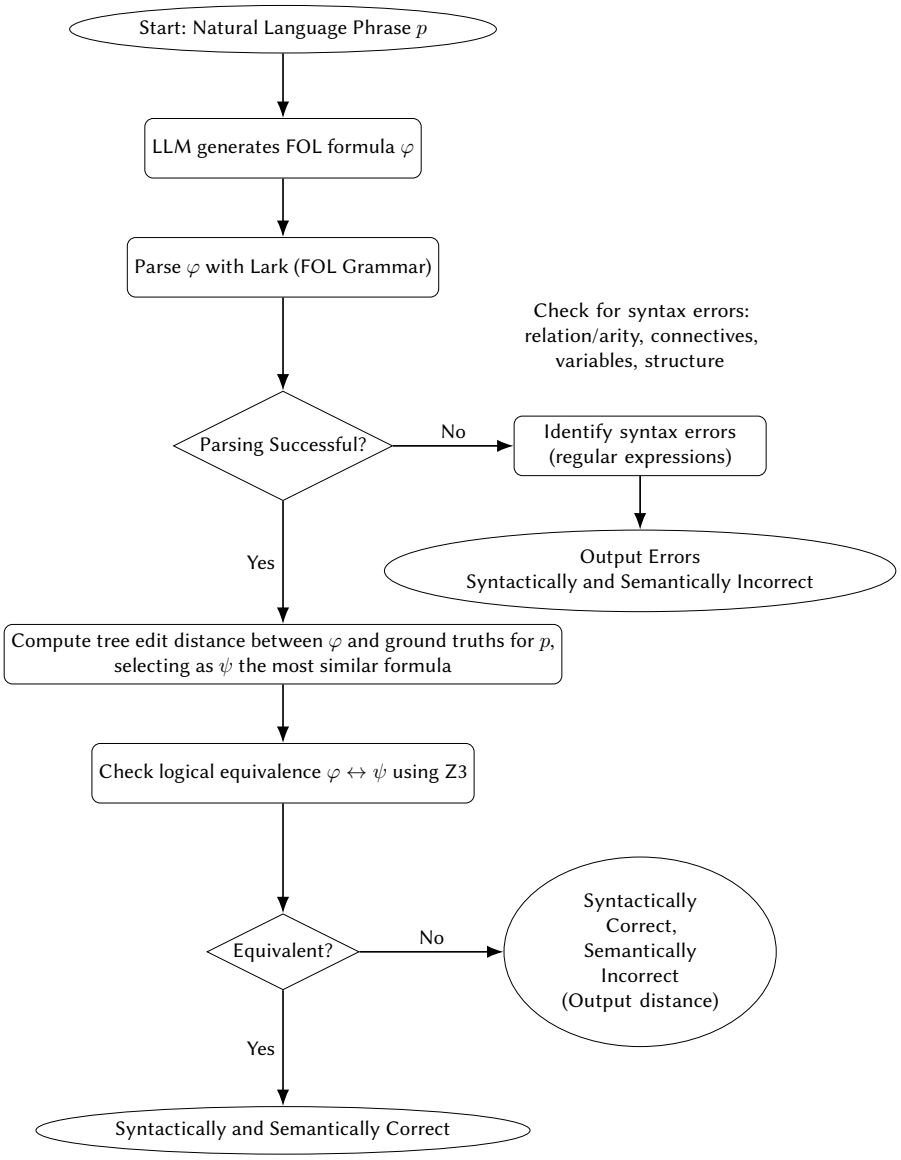

Figure 1: Overview of our NL-to-FOL evaluation pipeline

4. Experimental evaluation

In this section, we apply our pipeline and considered dataset to evaluate the LLMs Meta's Llama 3.1 (8B) and Google DeepMind's Gemma 2 (9B). To instruct the models to translate a natural language utterance, we used the DSPy framework [24] (Prompt 1). For the purpose of this preliminary study, we focus solely on the first attempts made by students, totaling 14,452,492 answers. This ensures a fair comparison with the LLMs, which, at present, are not involved in any form of iterative answering process.

Table 4 compares the accuracy of the LLMs in first-attempt formula generation with that of the average student. Overall, the LLMs achieve better results than the students. The comparison becomes more interesting when examining performance across different difficulty levels: while the average student's accuracy, of course, consistently decreases as the difficulty increases, the LLMs' performance is more varied. Llama 3.1 (8B) performs worse than the average student at the *super-low* and *low* difficulty levels, but outperforms at the higher levels. Gemma 2 (9B), while slightly surpassing LLama 3.1 overall, scores notably low at the *Mid-low* difficulty level.

Upon closer examination, it was found that among the 12.5% of formulas incorrectly generated by Llama 3.1 (8B), no exclusively semantic errors were made, as the pipeline had already exited at

```
Translate/Formalize English sentences into formulae of first-order language.

=== vocabulary to use ===
The vocabulary of the domain, part of a model for a first-order language, must be
    the following one:
Constants symbol: lower case character from a to f
Variables symbol: lower case character from s to z

=== Relations to use ===
Each element indicates the name of a relation and its arity (name/arity):
['Tet/1', 'Cube/1', 'Dodec/1', 'Small/1', 'Medium/1', 'Large/1', 'SameShape/2',
'SameSize/2', 'Larger/2', 'Smaller/2', 'SameCol/2', 'SameRow/2', 'Adjoins/2',
'FrontOf/2', 'RightOf/2', 'LeftOf/2', 'BackOf/2', 'Between/3']
Equality binary relation with symbol: =
Explanation of the relation Between:
    Between(x, y, z) means that x is between y and z
Explanation of the relation Larger:
    Larger(x, y) means that x is larger than y

=== symbols you MAY USE ===
Equality binary relation with symbol: =
Logical operator symbols: ¬, ∧, ∨ , → , ↔
Quantifier symbols: ∃, ∀

=== symbols you MUST NEVER USE ===
Symbols that are not allowed: "≠"

=== output constraint ===
The FOL formula is the first element of the output, with the prefix 'φ='.
```

Prompt 1: Instructions prompted to the LLMs to translate English sentences into FOL.

Table 4
LLMs and average student accuracy on first-attempt

	Super-low	Low	Mid-low	Mid-high	High	Super-high	Micro avg
Average student	94.90	86.85	76.47	62.11	57.65	47.35	85.99
Llama 3.1 (8B)	91.49	80.65	84.62	93.10	75.00	100.00	87.50
Gemma 2 (9B)	92.55	85.48	53.85	93.10	87.50	100.00	87.98

the syntax-checking stage. The distribution of syntax error types within these incorrect formulas is shown in Table 5 (columns labeled with 'L'). Despite the large percentage of uncategorized syntax errors (*Other*), it can be observed that the remaining errors generally involve the incorrect use of connectives or quantifiers. As for Gemma 2 (9B), exactly 3 formulas, belonging to the 12,02% of the incorrectly generated ones, passed the syntax check but failed the semantic test, with an average tree edit distance of 15. Looking again at Table 5 (columns labeled with 'G'), here the kinds of syntax errors detected are slightly more variegate, though errors involving connectives and quantifiers remain predominant.

Table 6 considers the agreement between the two LLMs. Notably, in 94.23% of cases, a phrase can be correctly formalized by at least one model, while in only 5.77% of cases neither model is able to produce a correct formula. For the 81.25% of the phrases, both models can derive the right formalization. Regarding the difficulty levels, it is noteworthy that, although the two LLMs share the same accuracy on the *Mid-high* level (93.10%, Table 4), they correctly formalize different sets of phrases.

Finally, in [12] it emerges that, for the students, the choice of the logical connective to be used in the formalization is a critical factor. Specific words, identified in both [12] and [15], have been found to be linked to this choice and are recognized as common sources of errors [12]. Therefore, in Table 7 we investigate whether LLMs make more or fewer mistakes than students when restricted to sentences containing words that are typically associated with errors. The accuracy distributions differ between the two LLMs and when compared to student results, suggesting that common sources of mistakes for students do not always apply to the models. Here, a notable case is that of *at least* (column 'AL').

Table 5
Distribution of syntax error types in incorrect LLMs generations, L: Llama 3.1 (8B), G: Gemma 2 (9B)

	Super-low		Low		Mid-low		Mid-high		High		Super-high	
	L	G	L	G	L	G	L	G	L	G	L	G
Relation/arity	0.00	0.00	0.00	33.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Connectives and quantifiers	62.50	85.71	33.33	33.33	50.00	50.00	0.00	50.00	50.00	100.00	0.00	0.00
Constants and variables	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Structure	0.00	0.00	8.33	0.00	0.00	16.67	0.00	50.00	50.00	0.00	0.00	0.00
Other	37.50	14.29	58.33	33.33	50.00	33.33	100.00	0.00	0.00	0.00	0.00	0.00

Table 6
Agreement (%) between LLMs, L: Llama 3.1 (8B), G: Gemma 2 (9B)

	Super-low	Low	Mid-low	Mid-high	High	Super-high	Micro avg
Either LLM correct	94.68	91.94	92.31	100.00	87.50	100.00	94.23
Both LLMs correct	89.36	74.19	46.15	86.21	75.00	100.00	81.25
No LLM correct	5.32	8.06	7.69	0.00	12.50	0.00	5.77
L correct, G incorrect	2.13	6.45	38.46	6.90	0.00	0.00	6.25
L incorrect, G correct	3.19	11.29	7.69	6.90	12.50	0.00	6.73

Table 7
LLMs and average student accuracy considering only sentences with words identified as sources of common mistakes [12, 15], AL: at least, EX: exactly, UN: unless, IFF: if and only if, OF: only if, ET: either, NET: neither, EV every, SM: some, ALL: all, SMT: something, JIC: just in case, * denotes words present in less than 5 phrases

	AL	EX*	UN*	IFF	OF	ET	NET	EV	SM	ALL	SMT	JIC*
Average student	74.95	76.79	78.18	83.7	78.13	86.85	83.94	85.36	89.34	87.38	79.49	84.45
LLama 3.1 (8B)	30.00	66.67	75.00	60.00	88.89	85.71	85.71	96.77	100.00	100.00	95.23	66.67
Gemma 2 (9B)	20.00	66.67	50.00	80.00	100.00	100.00	92.85	93.54	100.00	83.33	80.95	100.00

5. Discussion and Conclusions

In this work, we took an initial step toward defining a pipeline for evaluating LLMs on the task of formalizing natural language utterances into FOL formulas. We applied the pipeline to Meta's Llama 3.1 (8B) and Google DeepMind's Gemma 2 (9B), comparing their performance to that of students on the same tasks, using a large dataset based on Tarski's World. Already from our preliminary analyses, several interesting points emerged: (i) both LLMs slightly outperform the average student; (ii) there is no clear alignment between the phrases found difficult by the students and those deemed difficult by the LLMs; (iii) the vast majority of errors made by the LLMs are syntactic in nature; (iv) despite being of comparable size, the two LLMs exhibit different behaviors that, if strategically combined, could enhance translation performance; and (v) the common sources of mistakes for students differ from those of the considered LLMs.

Incidentally, these findings prompt important pedagogical questions, particularly concerning how the "thinking" in LLMs may fundamentally differ from student reasoning. For example, while LLMs rely on pattern recognition without an understanding of semantics, students learn through logical reasoning, which may indicate that LLM-generated feedback could be better suited for tasks with lower cognitive demands or for assisting below-average students rather than top-performing ones. Future studies could explore these distinctions further to assess whether LLMs can reliably assist students in tasks requiring deeper semantic comprehension or if they are more effective as support tools for students needing guidance with fundamental concepts. Additionally, it is worth investigating the specific ways in which the nature of LLM errors (primarily syntactic) contrasts with student errors, which could inform a targeted approach for LLM-based feedback that addresses common student mistakes more accurately.

Other than this, for future work, we plan to: extend our analyses to additional models and prompting

techniques; evaluate the alignment between accuracy improvements across repeated student submissions and multiple attempts by LLMs; investigate the (self-)correcting capabilities of LLMs; build on the notion of tree edit distance to define a metric for measuring the semantic distance between formulas; and conduct a more in-depth text analysis to uncover patterns and regularities related to formalization accuracy.

Acknowledgments

First and foremost, the authors thanks the invaluable support of David J. Barker-Plummer and John W. Etchemendy from Stanford University that provided us with the dataset to perform the experimentation. All the authors but RF and EM acknowledge the support from the 2024 Italian INdAM-GNCS project "Certificazione, monitoraggio, ed interpretabilità in sistemi di intelligenza artificiale", ref. no. CUP E53C23001670001. LG, AM, and NS also acknowledge the support from the Interconnected Nord-Est Innovation Ecosystem (iNEST), which received funding from the European Union Next-GenerationEU (PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR) – MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.5 – D.D. 1058 23/06/2022, ECS00000043). In addition, AM acknowledges the support from the MUR PNRR project FAIR - Future AI Research (PE00000013) also funded by the European Union Next-GenerationEU. Finally, EM and AM also acknowledge the support from the Autonomous region Friuli-Venezia Giulia for the project "Automazione Conversazionale e Voice&Speech Analytics per l'Active and Assisted Living" funded by the regional program FESR 2021-2027. This manuscript reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them.

References

[1] Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. *arXiv preprint arXiv:2012.13048*, 2020.

[2] Fei Yu, Hongbo Zhang, Prayag Tiwari, and Benyou Wang. Natural language reasoning, a survey. *ACM Computing Surveys*, 2023.

[3] Andrea Brunello, Luca Geatti, Angelo Montanari, and Nicola Saccomanno. Learning what to monitor: Using machine learning to improve past STL monitoring. In *Proceedings of the 33rd International Joint Conference on Artificial Intelligence*, pages 3270–3280. ijcai.org, 2024.

[4] John Stamper, Ruiwei Xiao, and Xinying Hou. Enhancing LLM-based feedback: Insights from intelligent tutoring systems and the learning sciences. In *Proceedings of the International Conference on Artificial Intelligence in Education*, pages 32–43. Springer, 2024.

[5] Andrea Brunello, Angelo Montanari, and Mark Reynolds. Synthesis of LTL formulas from natural language texts: State of the art and research directions. In *Proceedings of the 26th International symposium on temporal representation and reasoning*, pages 17:1–17:19. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

[6] Mohaimenul Azam Khan Raiaan, Md Saddam Hossain Mukta, Kaniz Fatema, Nur Mohammad Fahad, Sadman Sakib, Most Marufatul Jannat Mim, Jubaer Ahmad, Mohammed Eunus Ali, and Sami Azam. A review on large language models: Architectures, applications, taxonomies, open issues and challenges. *IEEE Access*, 2024.

[7] Yuan Yang, Siheng Xiong, Ali Payani, Ehsan Shareghi, and Faramarz Fekri. Harnessing the power of large language models for natural language to first-order logic translation. *arXiv preprint arXiv:2305.15541*, 2023.

[8] Abhinav Lalwani, Lovish Chopra, Christopher Hahn, Caroline Trippel, Zhijing Jin, and Mrinmaya Sachan. NL2FOL: Translating natural language to first-order logic for logical fallacy detection. *arXiv preprint arXiv:2405.02318*, 2024.

[9] Theo X Olausson, Alex Gu, Benjamin Lipkin, Cedegao E Zhang, Armando Solar-Lezama, Joshua B

Tenenbaum, and Roger Levy. LINC: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. *arXiv preprint arXiv:2310.15164*, 2023.

[10] Hanmeng Liu, Zhiyang Teng, Chaoli Zhang, and Yue Zhang. Logic agent: Enhancing validity with logic rule invocation. *arXiv preprint arXiv:2404.18130*, 2024.

[11] Dave Barker-Plummer, Richard Cox, and Robert Dale. Student translations of natural language into logic: The Grade Grinder corpus release 1.0. In *Proceedings of the 4th international conference on educational data mining*, pages 51–60, 2011.

[12] Dave Barker-Plummer, Richard Cox, Robert Dale, and John Etchemendy. An empirical study of errors in translating natural language into logic. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 30, 2008.

[13] Dave Barker-Plummer, Richard J. Cox, and Robert Dale. Dimensions of difficulty in translating natural language into first-order logic. *Educational Data Mining*, 2009.

[14] Dave Barker-Plummer, Robert Dale, Richard J. Cox, and Alex Romanczuk. Using edit distance to mine for errors in a natural language to logic translation corpus. *Educational Data Mining*, 2012.

[15] David Barker-Plummer, Jon Barwise, and John Etchemendy. *Language, Proof, and Logic: Second Edition*. Center for the Study of Language and Information/SRI, 2nd edition, 2011.

[16] Meta AI. Llama 3.1: Large language models for multilingual applications. https://www.llama.com/, 2024. Accessed: 2024-09-15.

[17] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[18] Team Gemma, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

[19] Dave Barker-Plummer, Jon Barwise, John Etchemendy, and Albert Liu. Tarski's World: Revised and expanded edition, 2007.

[20] Erez Shinan et al. Lark: A modern parsing library for python. https://github.com/lark-parser/lark, 2024. Accessed: 2024-09-05.

[21] Tushar Akhade et al. ZSS: A python library for tree edit distance. https://pythonhosted.org/zss/, 2024. Accessed: 2024-09-19.

[22] Kaizhong Zhang and Dennis Shasha. Simple fast algorithms for the editing distance between trees and related problems. *SIAM journal on computing*, 18(6):1245–1262, 1989.

[23] Leonardo de Moura et al. Z3 theorem prover. https://github.com/Z3Prover/z3, 2024. Accessed: 2024-09-07.

[24] Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, et al. DSPy: Compiling declarative language model calls into self-improving pipelines. *arXiv preprint arXiv:2310.03714*, 2023.