Scaling Synthetic Logical Reasoning Datasets with Context-Sensitive Declarative Grammars

Damien Sileo

Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 - CRIStAL, F-59000 Lille, France

damien.sileo@inria.fr

Abstract

Logical reasoning remains a challenge for natural language processing, but it can be improved by training language models to mimic theorem provers on procedurally generated problems. Previous work used domain-specific proof generation algorithms, which biases reasoning toward specific proof traces and limits auditability and extensibility. We present a simpler and more general declarative framework with flexible context-sensitive rules binding multiple languages (specifically, simplified English and the TPTP theorem-proving language). We construct first-order logic problems by selecting up to 32 premises and one hypothesis. We demonstrate that using semantic constraints during generation and careful English verbalization of predicates enhances logical reasoning without hurting natural English tasks. We use relatively small DeBERTa-v3 models to achieve state-of-the-art accuracy on the FOLIO human-authored logic dataset, surpassing GPT-4 in accuracy with or without an external solver by 12%.

1 Introduction

Language models trained only on natural language show lackluster capabilities at logical reasoning (McCoy et al., 2023; Mahowald et al., 2024). As a countermeasure, we can train neural models to match the output of symbolic reasoning systems (e.g., logic theorem provers, or other algorithms) on procedurally generated problems, to sharpen their reasoning capabilities. This process improves accuracy on some human-authored problems (Wu et al., 2021; Clark et al., 2020; Wu et al., 2022; Liu et al., 2023).

Previous work on synthetic first-order logic (FOL) reasoning datasets, RuleTaker (Clark et al., 2020), LogicNLI (Goodwin et al., 2020a) and FLD (Morishita et al., 2023), write dedicated code reimplementing the FOL axioms from scratch to generate proofs, and translate the generated problems to natural language. We propose Unigram, a framework for synthetic reasoning data generation, specifically designed to generate problems jointly into multiple languages. We represent grammars with concise and expressive rules binding two languages, and constraints to prune unwanted generations.

We write the most extensive grammar of FOL semantic fragments to our knowledge. We structure the generated expressions into PREMISE, HYPOTHESIS pairs, and annotate their logical relationship (entailment/contradiction/neutral) with a FOL solver, following the natural language inference (NLI) framework (Goodwin et al., 2020b). A simplistic FOL NLI problem is: PREMISE: *Everyone who is happy is rich. Mary is rich.* HYPOTHESIS: *Mary is happy* LABEL: *Neutral*.

We fine-tune DeBERTa NLI models (He et al., 2021) on Unigram-FOL and compare it with previous similar datasets. The 184M parameters (base-size) beats GPT-4 augmented or not with external theorem provers, on the FOLIO (Han et al., 2022) dataset. Our contributions are as follows: (i) A dataset of reasoning problems expressed in English and TPTP (a language that can be interfaced with numerous theorem provers) alongside Vampire proof annotations, covering FOL with equality and both finite and open domains, improved compositionality, and more extensive quantifiers. (ii) Ablations measuring the effect of constraining material conditionals usage, of using realistic English predicates, and of reimplementing LogicNLI with declarative generation instead of proof tree generation, highlighting that declarative can work better but that a richer logical modeling drives most of the improvement. (iii) A general reasoning problem grammar-based generation framework relying on solvers. The generation library, grammars, models, and generated dataset are publicly available[1].

[1] [code:GitHub][data:HF-datasets]

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 5275–5283
November 12-16, 2024

2 Related work

Synthetic datasets for reasoning Numerous works investigate the logical capabilities of NLP models using textual datasets and symbolic reasoning (Helwe et al., 2022). We focus on the grammar-derived synthetic datasets. RuleTaker (Clark et al., 2020) explores this area with a subset of first-order logic. LogicNLI addresses a broader FOL subset (Tian et al., 2021). FLD explores full FOL (Morishita et al., 2023) and increased compositionality. Richardson et al. (2020); Richardson and Sabharwal (2022) use a solver to study the satisfiability in natural language using the Z3 solver and dedicated generation logic on constrained problems. PrOntoQA (Saparov and He, 2023) generates proofs from ontologies and then derives questions from the proofs to analyze chains of thoughts in language models. Other work explore non-standard logic with synthetic dataset, notably probabilistic (Sileo and Moens, 2023), paraconsistant (Kazemi et al., 2024), epistemic (Sileo and Lernould, 2023) logics.

Generation frameworks Multiple frameworks already implement generation from handwritten grammars. NLTK (Bird and Loper, 2004) has a context-free grammar tool, but cannot natively handle multiple languages or large-scale generation. Grammatical Framework (Ranta, 2004) is the closest tool to ours. It enables generation from abstract grammars and linearization into concrete grammars (e.g. French and English) but it is translation-oriented and not context-sensitive. GLIF (Schaefer and Kohlhase, 2020) extends Grammatical Framework to parse English into logical formulas but is not suited for generation either.

3 Scalable dataset generation without forward inference

3.1 Forward inference

Previous NLI-style FOL reasoning datasets (RuleTaker, LogicNLI, FLD) generate examples using proof generators that are based on the axioms of FOL. This requires domain-specific generation code and introduces unwanted complexity. Elimination and Introduction rules can cancel each other and create an illusion of reasoning depth. We found that some examples in the Proofwriter dataset (Tafjord et al., 2021) directly contain the premise in the hypothesis despite having a reasoning depth of 5. When constructing NLI pairs, generating neutral examples requires special strategies introducing a sampling bias, and it can be the same for contradiction generation. Proof generation techniques enable high reasoning depth but at the cost of breadth (linguistic variety and reasoning variety).

3.2 Declarative generation

We fully rely on an existing FOL solver and we propose Unigram, a simpler, more generic method to generate problems with multilingual grammars where rules bind multiple surface form realization templates. A Unigram Rule declaration specifies a type signature, and two surface form realizers, and optional validity constraints: **R**(output_type, input_types, realizers, constraints) The signature specifies the type of the rule output and the type of the arguments. The realizers take the arguments as input and map them to a string. We can have a realizer for a logical representation and a realizer for English. Using functions allows more expressivity than context-free grammars (Hunter, 2021), but for most cases with can treat template strings as functions using Python `string.format`. Constraints and realizers can access the state of the current generation as an anytree tree. Constraints are binary functions checking construction validity. One useful constraint is distinctness, e.g. (arguments of the same type should have a different realization), to avoid repetitions or statements like *Mary likes Mary*. We enable this constraint by default.

Generation algorithm We use a depth-first algorithm that recursively fills in the leftmost non-terminal leaf with random type-matching rule sampling until constraints are satisfied. This enables left-to-right generation, allowing realizers and constraints to access the current context. We recursively call realizers to construct surface forms (e.g. English text).

4 Application to first-order logic (FOL)

We use Unigram to enrich FOL problem generation while also avoiding ambiguity, starting as a superset of LogicNLI (grammar in Appendix B). To create a problem, we uniformly sample 1 to 32 sentences as premises and 1 sentence per hypothesis ensuring that all symbols are present in the premise. We exclude non-satisfiable formulas (paradoxes) in premise groups and hypotheses. We label pairs as ENTAILMENT if $(premise \wedge \neg hypothesis)$ is

unsatisfiable, as CONTRADICTION if $(premise \wedge hypothesis)$ is unsatisfiable, and as NEUTRAL otherwise. Following Ruletaker and LogicNLI, we create problems with predicates over named individuals (e.g. *Mary is young*). We generate gender-balanced English surnames with CensusName. We now present new logical modeling features absent from the previous comparable datasets:

Explicit finite and open domains We explicitly mention the domain when using the quantifiers. We introduce two locations, *anywhere*, and a *room* with occupants e.g. *Mary, and Paul are the only persons in the room.* which logically means $\forall x, room(x) \rightarrow (x = \text{Mary} \vee x = \text{Paul})$. We can then quantify over the room (*everyone in the room*) or anywhere (*everyone anywhere*). By doing this, we can generate induction problems (checking that everyone in the room is happy if Mary and Paul are happy) and test reasoning with both finite and open domains. This requires handling FOL with equality which was not implemented in previous work.

Quantifiers and logical relationships We extend previous work with more complete quantifiers *not all, nobody, not everyone*. We leverage context-sensitivity to create a rule for polysyllogisms (predicate chains of the form all A are B, all B are C, all C are D. We also introduce *only if, unless, otherwise* as conditionals and allow sentence-level negation.

Constraining material conditionals Like previous work, we use material conditional to express conditional statements: *if p then q* is formalized as $p \rightarrow q$ i.e. $\neg p \vee q$. This means that the implication is true if p is false, and that negating $p \rightarrow q$ entails q both $\neg p$ which can be counter-intuitive. We use a constraint to eliminate all conditionals within the scope of negations and of other conditionals.

Improving predicate verbalization RuleTaker and LogicNLI use adjectives as logical predicates but do not handle their semantic interference. RuleTaker do not consider being both *blue* and being *green* as contradictory. LogicNLI uses 379 adjectives treated as independent, including *ugly* and *ugliest*. FLD uses pseudo language like *the lard does hurtle pushup*. We prompted GPT-4 (May version) to *Generate 150 predicates where each predicate does not contradict nor entail any other predicate. Two examples: "enjoys wildlife photography" and "owns a smart tv"*. We remove errors and provide manual negations. We also use relationships (*like, is a sibling of*, modeling symmetry axioms when relevant, and adjectives.

Logical representation language Previous LogicNLI, RuleTaker, FLD, and FOLIO all use their own logical format, representing formulas as lists or strings. We use the TPTP (Sutcliffe, 2010) FOF language which is a standard syntax for theorem provers evaluation and is compatible with many theorem provers, notably Vampire (Reger et al., 2022), Z3 (De Moura and Bjørner, 2008) or Prover9 (McCune, 2005). We select the Vampire (Reger et al., 2022) theorem prover which provides short and readable proofs and details all the premises used during a derivation.

Complexity control Methods based on forward inference can theoretically control the proof depth using hyperparameters. Here, to avoid mostly sampling shallow problems, we limit the number of non-neutral examples where the proof to the number of examples using 5 inputs, for each number of inputs. Neutrals are still a majority by an order of magnitude. To sample hard neutral examples, we use a Gradient Boosting classifier with 100 trees (and scikit-learn 1.5.0 (Pedregosa et al., 2011) default parameters otherwise) to predict the labels based on unigram counts of the logical operators in the premise and hypothesis. We train on $1k$ examples, discard these, and then discard the most confident neutral predictions to achieve balanced labels.

5 Experiments

5.1 Methodology

We fine-tune a pre-trained NLI model on multiple synthetic FOL datasets: LogicNLI, FLD, RuleTaker, and on Unigram-FOL. We then evaluate the direct effect on other three-way entailment downstream tasks, and on further fine-tuning on the training data of evaluation tasks (Phang et al., 2018).

We use the DeBERTa-v3 (He et al., 2021) NLI models trained on the tasksource collection (Sileo, 2024)[2]. We use a learning rate of $1e{-}5$ for DeBERTa-large and $2e{-}5$ (Mosbach et al., 2021) for DeBERTa-base, 1 or 3 epochs (based on intrinsic validation accuracy) and Huggingface Transformers (Wolf et al., 2019) version 4.41 default Trainer arguments otherwise.

[2] hf.co/deberta-v3-base-tasksource-nli

Model size	Auxilary training	FOLIO	+ft	WANLI	+ft	CTRL	+ft	Fragments
D-base	-	49.5	74.3	**65.2**	**77.4**	**46.2**	**56.7**	63.6
D-base	RuleTaker	55.1	71.3	60.9	73.8	36.0	53.0	48.7
D-base	LogicNLI	50.5	69.3	61.1	72.4	38.4	54.4	56.3
D-base	FLD	59.9	72.3	60.0	73.6	38.2	55.8	56.8
D-base	Unigram-FOL	**64.4**	**78.2**	63.6	75.6	42.8	56.6	**65.4**
D-base	– Constrained_Conditionals	63.4	81.2	62.2	71.8	40.6	55.4	59.8
D-base	– Realistic_Predicates	62.4	76.2	65.8	74.4	41.8	53.2	68.2
D-base	Unigram-LogicNLI	57.4	71.3	61.6	76.4	38.6	55.6	57.8
D-large	-	49.5	70.0	66.2	77.0	**49.6**	62.0	67.6
D-large	RuleTaker	58.1	77.2	68.5	77.9	43.1	60.7	61.7
D-large	LogicNLI	58.7	73.3	68.5	77.4	45.4	60.9	64.4
D-large	FLD	60.9	78.2	68.0	77.6	44.0	59.8	61.7
D-large	Unigram-FOL	**63.4**	**82.2**	**75.4**	**81.6**	48.2	**62.2**	**73.2**
D-large	Unigram-FOL+FLD	**78.2**	**88.6**	65.2	78.4	42.2	57.9	**75.4**

Figure 1: Comparison of auxiliary synthetic training datasets effect on the evaluation tasks. We report the average accuracy of two runs. $\mathcal{D}$ column refer to zero-shot $\mathcal{D}$ test accuracy after synthetic auxiliary training, and +ft refers to the test accuracy after auxiliary training then further fine-tuning $\mathcal{D}$ training set (in the previous column).

We generate $100k$ examples with a $80/10/10$ train/dev/test split. but we only use $40k$ training training examples to match FLD. We use the FLD$\star$ version of FLD. We use the ProofWriter (Tafjord et al., 2021) open-world-assumption version of RuleTaker. We exclude LogicNLI examples labeled as paradoxes and we map all labels to NLI labels.

5.2 Evaluation datasets

We evaluate on two pure reasoning datasets, FOLIO and Fragments, and on two more general datasets: **FOLIO** (Han et al., 2022) contains human-written FOL problems. We evaluate on the validation set to compare to Olausson et al. (2023) results who report 72.5% accuracy using a GPT-4 with a solver and 75.3% with chain-of-thoughts. We construct another validation set from 10% of train and map labels to NLI labels. (Wei et al., 2022) **WANLI** (Liu et al., 2022) is a NLI dataset with diverse and challenging reasoning patterns. **ConTRoL** (Liu et al., 2021) is a NLI dataset requiring multiple premises to derive the correct label, measuring contextual reasoning. **Fragments** (Richardson et al., 2020) is based on formal semantics templates and evaluate reasoning with quantifiers; this dataset is mostly suited to evaluation, as training quickly leads to almost perfect test accuracy.

Comparison with previous synthetic datasets Table 1 shows the accuracy of multiple auxiliary training datasets on the evaluation dataset. Unigram-FOL outperforms RuleTaker, LogicNLI, and FLD on all tasks with a comfortable margin, and leads to lesser degradation on the datasets that are not only focused on logic (WANLI, ConTRoL). The last line of the table combines Unigram-FOL (with the full 100k examples) with FLD and shows that combining generation methods can further push the state of the art on FOLIO.

We conduct ablations to better understand the source of this improvement, presented in the middle of Table 1.

Unigram-LogicNLI We use our declarative generation method on the base LogicNLI grammar to disentangle the effect of the generation technique from the grammar itself. This outperforms the original LogicNLI but not Unigram-FOL which highlights the value of our additional constructions.

Replacing Realistic Predicates We replace our generated predicates with the original LogicNLI adjectives (containing semantic interferences); this degrades FOLIO accuracy but does not strongly impact other NLI tasks, notably Fragments which mainly use adjectives as predicates.

Removing Conditionals Constraints Unrestricting usage of material conditionals harms the zero-shot transfer on FOLIO and the capabilities at more general reasoning, which confirms that removing counter-intuitive constructs can help transferability.

6 Conclusion

We showed that simple declarative grammars paired with solvers can outperform complex proof tree generators for reasoning dataset generations and released a new FOL reasoning dataset, models, and ablations. Our framework can help future reasoning research, notably on explanation since fully aligned TPTP code can be leveraged to model necessity and sufficiency. We plan to extend Unigram to planning, constraint satisfaction and modal logic.

Limitations

Reasoning methods based on neural networks do not provide formal guarantees and can introduce biases in real applications. They can be used as a complement to externalization methods (Olausson et al., 2023). Automatically formalizing a problem is difficult and can lead to mistakes (Olausson et al., 2023) which could be detected by internalization-based methods. Our dataset could be used to automate formalization but we did not try such experiments. In addition, our work is only conducted with English language and encoder models, mainly used for verification and not generation. We only used one model architecture, DeBERTa, while other architectures like Albert (Lan et al., 2020) or other recursive architectures could be more suited to reasoning.

Ethical considerations

Our models are derived from language models which inherit bias from their training corpus. We did not conduct any human annotations, relying on already annotated datasets to validate our methodology. We use encoder models which have lower energy consumption than decoders (Luccioni et al., 2024) and performed experiments with less than 20 total days on a Nvidia A100 GPU.

References

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2020. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3882–3890. International Joint Conferences on Artificial Intelligence Organization. Main track.

Leonardo De Moura and Nikolaj Bjørner. 2008. Z3: An efficient smt solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340. Springer.

Emily Goodwin, Koustuv Sinha, and Timothy J O'Donnell. 2020a. Probing linguistic systematicity. *arXiv preprint arXiv:2005.04315*.

Emily Goodwin, Koustuv Sinha, and Timothy J. O'Donnell. 2020b. Probing linguistic systematicity. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1958–1969, Online. Association for Computational Linguistics.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Luke Benson, Lucy Sun, Ekaterina Zubova, Yujie Qiao, Matthew Burtell, et al. 2022. Folio: Natural language reasoning with first-order logic. *arXiv preprint arXiv:2209.00840*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Chadi Helwe, Chloé Clavel, and Fabian Suchanek. 2022. Logitorch: A pytorch-based library for logical reasoning on natural language. In *The 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

Tim Hunter. 2021. The chomsky hierarchy. *A companion to Chomsky*, pages 74–95.

Mehran Kazemi, Quan Yuan, Deepti Bhatia, Najoung Kim, Xin Xu, Vaiva Imbrasaite, and Deepak Ramachandran. 2024. Boardgameqa: A dataset for natural language reasoning with contradictory information. *Advances in Neural Information Processing Systems*, 36.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Alisa Liu, Swabha Swayamdipta, Noah A. Smith, and Yejin Choi. 2022. WANLI: Worker and AI collaboration for natural language inference dataset creation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6826–6847, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Hanmeng Liu, Leyang Cui, Jian Liu, and Yue Zhang. 2021. Natural language inference in context - investigating contextual reasoning over long texts. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15):13388–13396.

Qian Liu, Fan Zhou, Zhengbao Jiang, Longxu Dou, and Min Lin. 2023. From zero to hero: Examining the power of symbolic tasks in instruction tuning. *arXiv preprint arXiv:2304.07995*.

Sasha Luccioni, Yacine Jernite, and Emma Strubell. 2024. Power hungry processing: Watts driving the cost of ai deployment? In *The 2024 ACM Conference on Fairness, Accountability, and Transparency*, pages 85–99.

Kyle Mahowald, Anna A Ivanova, Idan A Blank, Nancy Kanwisher, Joshua B Tenenbaum, and Evelina Fedorenko. 2024. Dissociating language and thought in large language models. *Trends in Cognitive Sciences*.

R Thomas McCoy, Shunyu Yao, Dan Friedman, Matthew Hardy, and Thomas L Griffiths. 2023. Embers of autoregression: Understanding large language models through the problem they are trained to solve. *arXiv preprint arXiv:2309.13638*.

William McCune. 2005. Release of prover9. In *Mile high conference on quasigroups, loops and nonassociative systems, Denver, Colorado*.

Terufumi Morishita, Gaku Morio, Atsuki Yamaguchi, and Yasuhiro Sogawa. 2023. Learning deductive reasoning from synthetic corpus based on formal logic. In *International Conference on Machine Learning*, pages 25254–25274. PMLR.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning {bert}: Misconceptions, explanations, and strong baselines. In *International Conference on Learning Representations*.

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.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Aarne Ranta. 2004. Grammatical framework. *Journal of Functional Programming*, 14(2):145–189.

Giles Reger, Martin Suda, Andrei Voronkov, Laura Kovács, Ahmed Bhayat, Bernhard Gleiss, Marton Hajdu, Petra Hozzova, JR Evgeny Kotelnikov, Michael Rawson, et al. 2022. Vampire 4.7-smt system description.

Kyle Richardson, Hai Hu, Lawrence Moss, and Ashish Sabharwal. 2020. Probing natural language inference models through semantic fragments. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:8713–8721.

Kyle Richardson and Ashish Sabharwal. 2022. Pushing the limits of rule reasoning in transformers through natural language satisfiability. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11209–11219.

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*.

Jan Frederik Schaefer and Michael Kohlhase. 2020. Glif: A declarative framework for symbolic natural language understanding. In *FCR@ KI*, pages 4–11.

Damien Sileo. 2024. tasksource: A large collection of nlp tasks with a structured dataset preprocessing framework. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 15655–15684.

Damien Sileo and Antoine Lernould. 2023. MindGames: Targeting theory of mind in large language models with dynamic epistemic modal logic. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4570–4577, Singapore. Association for Computational Linguistics.

Damien Sileo and Marie-francine Moens. 2023. Probing neural language models for understanding of words of estimative probability. In *Proceedings of the 12th Joint Conference on Lexical and Computational Semantics (*SEM 2023)*, pages 469–476, Toronto, Canada. Association for Computational Linguistics.

Geoff Sutcliffe. 2010. The tptp world–infrastructure for automated reasoning. In *International Conference on Logic for Programming Artificial Intelligence and Reasoning*, pages 1–12. Springer.

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.

Jidong Tian, Yitian Li, Wenqing Chen, Liqiang Xiao, Hao He, and Yaohui Jin. 2021. Diagnosing the first-order logical reasoning ability through LogicNLI. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3738–3747, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou,

et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yuhuai Wu, Felix Li, and Percy S Liang. 2022. Insights into pre-training via simpler synthetic tasks. *NEURIPS 2022*, 35:21844–21857.

Yuhuai Wu, Markus N Rabe, Wenda Li, Jimmy Ba, Roger B Grosse, and Christian Szegedy. 2021. Lime: Learning inductive bias for primitives of mathematical reasoning. In *ICML*, pages 11251–11262. PMLR.

A FOL-nli example

PREMISE :
Christopher, Donald, Gene are the only persons in the room.
Everyone in the room who collects antique jewelry plays the drums.
Someone in the room designs and sews custom cosplay costumes for conventions.
Christopher collects classic novels.
Everyone in the room who enjoys deep-sea diving and exploring underwater caves enjoys kayaking or is a night owl or both.
Christopher enjoys kayaking.
Everyone in the room enjoys kayaking only if they collects antique jewelry.
HYPOTHESIS :
Christopher collects antique jewelry.
LABEL
entailment
PREMISE (TPTP):

```
room(c) & room(d) & room(g) &
(![X]:(room(X) => (X='c' | X='d' | X='g'))) &
(![X]:(room(X) => ((collects_jewelry(X)) => (plays_drums(X))))) &
(?[X]:(room(X) & (designs_cosplay(X)))) &
collects_novels(c) &
(![X]:(room(X) => ((enjoys_diving(X)) =>
(enjoys_kayaking(X) | is_night_owl(X))))) &
enjoys_kayaking(c) &
(![X]:(room(X) => (enjoys_kayaking(X) <= collects_jewelry(X))))
```

B Unigram LogicNLI reimplementation

```
from unigram import Rule as R


ADJECTIVES = ['rich','quiet','old','tall','kind','brave','wise',
'happy', 'strong','curious','patient','funny','generous','humble']
# (We selected adjectives with no clear semantic interference)
NAMES = ['mary', 'paul', 'fred', 'alice', 'john', 'susan', 'lucy']

R.init(['tptp','eng'], "fof")

R('premise(' + ','.join(['rule']*16')'+','+'.'join(['fact]*8)+')',
  '&\n'.join([f'({i})' for i in range(24)]),
   '\n'.join([f'{i}' for i in range(24)]))

R('hypothesis(person,a)','1(0)','0 is 1')

for a in ADJECTIVES:
    R('adj', a), R('adj', f'~{a}', f'not {a}', weight=0.2)

R('property(adj,adj)', '(0(?)&1(?))', 'both 0 and 1')
R('property(adj,adj)', '(0(?)|1(?))', '0 or 1')
R('property(adj,adj)', '(0(?)<~>1(?))', 'either 0 or 1', weight=0.5)
R('property(adj)', '0(?)', '0')

R('rule(property,property)', '![X]:(0[?←X]=>1[?←X])',
'everyone who is 0 is 1')
R('rule(property,property)', '![X]:(0[?←X]<=>1[?←X])',
'everyone who is 0 is 1 and vice versa')

for p in NAMES:
    R('person', p)

R('fact(person,property)','1[?←0]', '0 is 1')
R('fact(property)', '?[X]:(0[?←X])', 'someone is 0', weight=0.2)

R('rule(fact,fact)', '(0)=>(1)', 'if 0 then 1')
R('rule(fact,fact)', '(0)<=>(1)', 'if 0 then 1 and vice versa')
```