LAMBADA: BACKWARD CHAINING FOR AUTOMATED REASONING IN NATURAL LANGUAGE

Anonymous authors
Paper under double-blind review

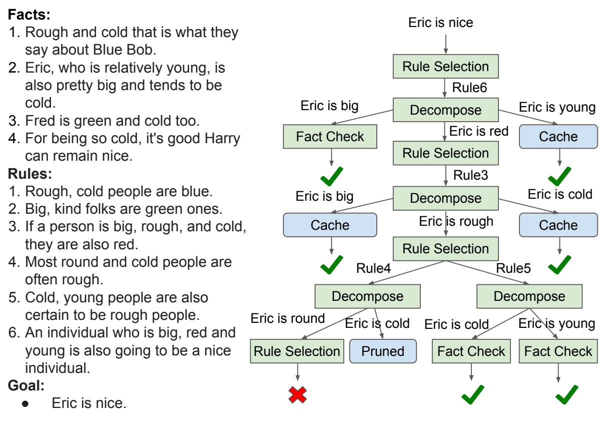
ABSTRACT

Remarkable progress has been made on automated reasoning with natural text, by using Large Language Models (LLMs) and methods such as Chain-of-Thought prompting and Selection-Inference. These techniques search for proofs in the forward direction from axioms to the conclusion, which suffers from a combinatorial explosion of the search space, and thus high failure rates for problems requiring longer chains of reasoning. The classical automated reasoning literature has shown that reasoning in the backward direction (i.e. from the intended conclusion to supporting axioms) is significantly more efficient at proof-finding. Importing this intuition into the LM setting, we develop a neuro-symbolic *Backward Chaining* algorithm, called LAMBADA, that decomposes reasoning into four sub-modules, that are simply implemented by few-shot prompted LLM inference. We show that LAMBADA achieves sizable accuracy boosts over state-of-the-art forward reasoning methods on two challenging logical reasoning datasets, particularly when deep and accurate proof chains are required.

1 INTRODUCTION

Automated reasoning, the ability to draw valid conclusions from explicitly provided knowledge, has been a fundamental goal for AI since its early days (McCarthy, 1959; Hewitt, 1969). Furthermore, logical reasoning, especially reasoning with unstructured, natural text is an important building block for automated knowledge discovery and holds the key for future advances across various scientific domains. While in recent years tremendous progress has been made towards natural language understanding thanks to pre-trained language models (LMs) (Brown et al., 2020; Chowdhery et al., 2022), the performance of these models for logical reasoning still lags behind (Rae et al., 2021; Creswell et al., 2022; Valmeekam et al., 2022) compared to the advancements in other areas such as reading comprehension and question-answering.

While many problems benefit from LM scaling, scaling has been observed to provide limited benefit for solving complex reasoning problems. For example, Creswell et al. (2022) observed that for the Gopher family of LMs (Rae et al., 2021), the scaling law for logic-based tasks is significantly worse than for other language tasks. Moreover, while finetuning initially seemed to enable logical reasoning in LMs (Clark et al., 2020; Tafjord et al., 2021), further exploration revealed that finetuned LMs mostly exploit spurious correlations (e.g., the correlation between the number of rules and the label) as opposed to learning to reason (Zhang et al., 2022; Schlegel et al., 2022; Liu et al., 2022). Recently, prompting strategies such as Chain-of-Thought (Wei et al.,


Facts:
1. Rough and cold that is what they say about Blue Bob.
2. Eric, who is relatively young, is also pretty big and tends to be cold.
3. Fred is green and cold too.
4. For being so cold, it's good Harry can remain nice.

Rules:
1. Rough, cold people are blue.
2. Big, kind folks are green ones.
3. If a person is big, rough, and cold, they are also red.
4. Most round and cold people are often rough.
5. Cold, young people are also certain to be rough people.
6. An individual who is big, red and young is also going to be a nice individual.

Goal:
- Eric is nice.

Figure 1: The search trace of LAMBADA on an example from ParaRules (the *Sign Agreement* and failed *Fact Check* modules are omitted for brevity).

2022) and Scratchpad (Nye et al., 2022) have found some success for such tasks, although they have been also shown to struggle with proof planning for more complex logical reasoning problems (Saparov & He, 2022).

One solution to the aforementioned problems is a neuro-symbolic approach that integrates the strength and reliability of classical AI models in logical reasoning with LMs (Garcez & Lamb, 2020; Marcus, 2020). In the classic literature, there are two major approaches to logical reasoning (Poole & Mackworth, 2010):

1. *Forward Chaining (FC)* where one starts from the facts and rules ("theory"), and iterates between making new inferences and adding them to the theory until the goal statement can be proved or disproved,
2. *Backward Chaining (BC)* where one starts from the goal and uses the rules to recursively decompose it into sub-goals until the sub-goals can be proved or disproved based on the facts.

Previous approaches to reasoning with LMs mostly incorporate elements of FC into LMs (Tafjord et al., 2021; Creswell et al., 2022). FC requires selecting a subset of facts and rules from the entire theory which might be difficult for an LM as it requires a combinatorial search over a large space. Moreover, deciding when to halt and declare failure to prove is challenging in FC (Creswell et al., 2022), sometimes requiring specialized modules trained on intermediate labels (Creswell & Shanahan, 2022). Indeed, the classic automated reasoning literature is heavily weighted towards BC or goal-directed strategies for proof-finding.

In this paper, we argue and show experimentally that BC is better suited for text-based deductive logical reasoning, as it does not require a combinatorial search for subset selection and there are more natural halting criteria for it. We develop a hybrid **LA**nguage **M**odel augmented **BA**ckwar**D** ch**A**ining technique (LAMBADA), where BC drives the high-level proof planning, and the LM performs the textual understanding and individual reasoning steps. We conduct experiments with challenging datasets for LM reasoning containing examples with natural text, requiring proof chains of up to 5 hops in length, and examples where the goal can neither be proved nor disproved from the provided theory. We show that LAMBADA achieves substantially higher deductive accuracy, and is considerably more likely to generate valid reasoning chains compared to other techniques which find correct conclusions with spurious proof traces, while also being more query efficient than other LM-based modular reasoning approaches. Our results strongly indicate that future work on reasoning with LMs should incorporate backward chaining or goal-directed strategies.

2 LAMBADA: LANGUAGE MODEL AUGMENTED BACKWARD CHAINING

A theory consists of a set of facts and a set of rules. We focus on performing automated reasoning over natural language assertions such as `''Nice people are red''` that are coherent but not necessarily grounded in reality. An example theory with fictional characters and rules is demonstrated in Figure 1. Based on the theory, one may want to prove the goal `''Eric is nice''`.

Algorithm 1 LAMBADA

Input: Theory $\mathcal{C} = (\mathcal{F}, \mathcal{R})$, Goal $\mathcal{G}$, Max-Depth D

```
1:  factCheckResult = FactCheck($\mathcal{G}$, $\mathcal{F}$)
2:  if factCheckResult $\neq$ UNKNOWN then
3:      return factCheckResult
4:  if D == 0 then
5:      return UNKNOWN
6:  $\mathcal{R}_s$ = RuleSelection($\mathcal{G}$, $\mathcal{R}$)
7:  for $r \in$ Rerank($\mathcal{R}_s$) do
8:      G = GoalDecomposition($r$, $\mathcal{G}$)
9:      if ProveSubgoals($\mathcal{C}$, G, D) then
10:         if SignAgreement(r, $\mathcal{G}$) then
11:             return PROVED
12:         else
13:             return DISPROVED
14: return UNKNOWN
```

Figure 1 also shows an example of backward chaining (BC) applied to a theory to prove a goal. Initially, BC verifies if the goal can be proved or disproved based on the facts (this step is omitted from the figure). Since none of the facts prove or disprove the goal, BC next selects a rule (Rule6) that can be applied to break down the goal into sub-goals. Applying Rule6 breaks down the goal into three sub-goals. BC makes recursive calls to prove each sub-goal. The algorithm continues until either a halting criterion is reached (e.g., reaching a certain depth in search), or a sub-goal can no longer be broken down (e.g., the left sub-tree under `''Eric is rough''`), or all sub-goals are proved (e.g., the right sub-tree under `''Eric is rough''`). Depending on the theory and the goal, the output of BC is either PROVED, DISPROVED, or UNKNOWN.

2.1 THE LAMBADA ALGORITHM

To enable applying BC for text-based reasoning, we introduce four LM-based modules: *Fact Check* (given a set of facts and a goal predicts if the goal can be proved, disproved, or neither), *Rule Selection* (given a set of rules and a goal, determines which rules can be applied to break the goal into sub-goals), *Goal Decomposition* (given a rule and goal, breaks down the goal into sub-goals based on the rule), and *Sign Agreement* (given a goal and a rule, predicts if the sign of the goal agrees with the sign of the consequent of the rule). Each module is implemented by showing relevant in-context demonstrations to a pretrained LM. Algorithm 1 provides a high-level description of how these modules can be integrated with BC to enable text-based logical reasoning (the function calls corresponding to LM modules are color-coded).

Algorithm 2 ProveSubgoals

```
Input: Theory $\mathcal{C} = (\mathcal{F}, \mathcal{R})$, Sub-Goals $\mathbf{G}$, Max-Depth D
1: for $\mathcal{G}$ in $\mathbf{G}$ do
2:     result = LAMBADA($\mathcal{C}$, $\mathcal{G}$, D-1)
3:     if result $\neq$ PROVED then
4:         return False # Assuming conjunction
   return True
```

LAMBADA can be understood as a depth-first search algorithm over the facts and the rules. It takes as input a theory $\mathcal{C} = (\mathcal{F}, \mathcal{R})$, a goal $\mathcal{G}$, and a depth D that defines a halting criterion based on the maximum allowed depth for the search. Initially, the algorithm uses the *Fact Check* module to check if $\mathcal{G}$ can be proved or disproved using the facts. If this is the case, the algorithm stops and returns the result. Otherwise, the algorithm checks the depth D: if $D = 0$, then the algorithm stops and returns UNKNOWN. Otherwise, the algorithm proceeds with applying rules. The *Rule Selection* module is used to identify the rules $\mathcal{R}_s$ from $\mathcal{R}$ whose consequent unifies with $\mathcal{G}$. For each selected rule, the algorithm uses the *Goal Decomposition* module to decompose $\mathcal{G}$ into a set of sub-goals $\mathbf{G}$ that need to be proved and checks whether those sub-goals can be proved by making recursive calls to the algorithm (with reduced depth). If the sub-goals can be proved, then the algorithm uses the *Sign Agreement* module to check whether the sign of the rule consequent agrees or disagrees with the sign of $\mathcal{G}$. If it does, then the algorithm returns PROVED and otherwise DISPROVED. If there is no rule for which the sub-goals can be proved, then UNKNOWN is returned.

3 EXPERIMENTS AND RESULTS

All experiments are based on the PaLM 540B language model (Chowdhery et al., 2022).

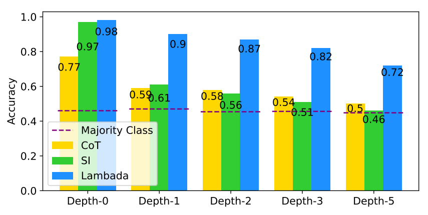
Baselines: We compare against two baselines: Chain of Thought (CoT) (Wei et al., 2022), a popular neural approach based on demonstrating chains of inference to the LM within the in-context prompt, and Selection-Inference (SI) (Creswell et al., 2022), a strong modular reasoning approach.

Datasets: We experiment with challenging logical reasoning datasets namely **ProofWriter** (Tafjord et al., 2021), **PrOntoQA** (Saparov & He, 2022) and **ParaRules** (Tafjord et al., 2021). ProofWriter and PrOntoQA contain sub-categories based on the (maximum) number of reasoning hops required. We report results for each sub-category separately. ParaRules is a version of ProofWriter where the theory is rewritten by crowdworkers for increased diversity and naturalness (see an example if Figure 1). To reduce experimentation cost, for ProofWriter we use the first 1000 examples from the test set (OWA category) and for ParaRules we use the first 500 examples[1].

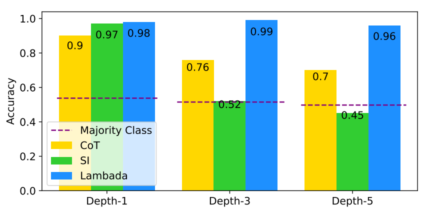
Label Accuracy The results are reported in Figure 2(a)-(d).[2] We observe that LAMBADA significantly outperforms the baselines, especially on ProofWriter-PUD which contains UNKNOWN labels (44% relative improvement compared to CoT and 56% compared to SI on Depth-5), the higher depths of PrOntoQA (37% relative improvement compared to CoT and 113% compared to SI on Depth-5), and the ParaRules dataset (43% relative improvement compared to CoT). These results overall show the merit of LAMBADA for logical reasoning. Furthermore, we highlight that the reasoning capacity of LAMBADA robustly generalizes to more naturalistic expressions, as demonstrated by the high accuracy on ParaRules, which is exactly the desired outcome of combining the strengths of an LM and a reasoning algorithm.

[1]ParaRules had some minor quality issues that we manually fixed before experimentation.

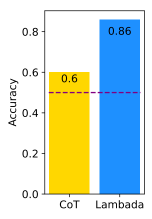
[2]Due to the low performance of SI on the other datasets and its high number of LM calls, for ParaRules we only compared LAMBADA against CoT.



(a)

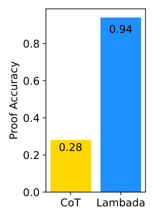


(b)

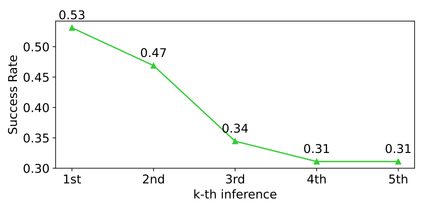


(c)

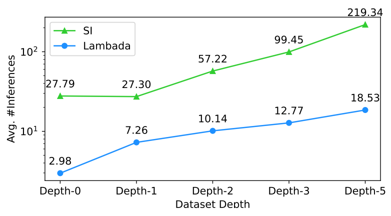
Figure 2: Prediction accuracy results on (a) ProofWriter (b) PrOntoQA, and (c) ParaRules datasets.



(a)



(b)



(c)

Figure 3: (a) Proof accuracy comparison. (b) Forward chaining becomes progressively harder. (c) LAMBADA is more inference call efficient than SI.

Proof Accuracy: We compared LAMBADA to CoT in terms of proof accuracy when the label is predicted correctly. To this end, we randomly selected 50 examples from depth-5 of the ProofWriter dataset where each model predicted the label correctly and we manually verified the proofs. The results in Figure 3(a). show that LAMBADA produces substantially more accurate proofs than CoT.

Inefficacy of Forward Chaining: SI is based on forward chaining and its selection module requires a combinatorial search to find the right subset of facts and rules, and the search space becomes progressively larger in each iteration of the algorithm as new inferences are added to the theory. To verify whether the increase in the search space makes forward chaining progressively harder, we measured the success rate of the k-th inference of SI for different values of k on Depth-5 of PrOntoQA. According to the results in Figure 3(b), the success rate indeed decreases significantly in the later inferences when the size of the theory (and hence the search space) increases. We also find SI (and more generally) suffer severely from duplicate inference generation where running the inference multiple times results in producing the same inference over and over again.

Number of Inference Calls: Another advantage of LAMBADA is its comparative efficiency compared to other approaches that require multiple inference calls to the LM. In Figure 3(c), we compare the average number of LM inference calls made per example, for different depths of ProofWriter. We observe that LAMBADA requires significantly fewer calls, especially at higher depths.

Qualitative Analysis: In Figure 1, we show the search trace created by LAMBADA for an example from ParaRules, where the answer was predicted correctly. From the figure, one can see how backward chaining helps LAMBADA effectively search and create the reasoning chain and how the LM helps fact checking, rule selection, goal decomposition, and sign agreement checking.

Conclusion: We developed LAMBADA, a neuro-symbolic algorithm for reasoning in natural language that combines the capacity of LMs to handle naturalistic text with the backward chaining algorithm for high-level reasoning. We showed that LAMBADA achieves significant improvements over competitive approaches on challenging benchmarks. We believe our key insight on the efficacy of neuro-symbolic reasoning with bacward chaining and LMs is widely applicable and can be adapted to other NLP tasks where multi-step inference may be required.

REFERENCES

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. PaLM: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. Transformers as soft reasoners over language. *arXiv preprint arXiv:2002.05867*, 2020.

Antonia Creswell and Murray Shanahan. Faithful reasoning using large language models. *arXiv preprint arXiv:2208.14271*, 2022.

Antonia Creswell, Murray Shanahan, and Irina Higgins. Selection-inference: Exploiting large language models for interpretable logical reasoning. *arXiv preprint arXiv:2205.09712*, 2022.

Artur d'Avila Garcez and Luis C Lamb. Neurosymbolic ai: the 3rd wave. *arXiv preprint arXiv:2012.05876*, 2020.

Carl Hewitt. Planner: A language for proving theorems in robots. In *Proceedings of the 1st International Joint Conference on Artificial Intelligence*, IJCAI'69, pp. 295–301, San Francisco, CA, USA, 1969. Morgan Kaufmann Publishers Inc.

Bingbin Liu, Jordan T Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Transformers learn shortcuts to automata. *arXiv preprint arXiv:2210.10749*, 2022.

Gary Marcus. The next decade in ai: four steps towards robust artificial intelligence. *arXiv preprint arXiv:2002.06177*, 2020.

John McCarthy. Programs with common sense. In *Proceedings of the Teddington Conference on the Mechanization of Thought Processes*, pp. 75–91, London, 1959. Her Majesty's Stationary Office. URL `http://www-formal.stanford.edu/jmc/mcc59.html`.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. Show your work: Scratchpads for intermediate computation with language models. In *Deep Learning for Code Workshop*, 2022. URL `https://openreview.net/forum?id=HBlx2idbkbq`.

David L Poole and Alan K Mackworth. *Artificial Intelligence: foundations of computational agents*. Cambridge University Press, 2010.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

Abulhair Saparov and He He. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. *arXiv preprint arXiv:2210.01240*, 2022.

Viktor Schlegel, Kamen V Pavlov, and Ian Pratt-Hartmann. Can transformers reason in fragments of natural language? *arXiv preprint arXiv:2211.05417*, 2022.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3621–3634, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.317. URL `https://aclanthology.org/2021.findings-acl.317`.

Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. Large language models still can't plan (a benchmark for llms on planning and reasoning about change). *arXiv preprint arXiv:2206.10498*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang, and Guy Van den Broeck. On the paradox of learning to reason from data. *arXiv preprint arXiv:2205.11502*, 2022.