COMPOSITIONAL NEURO-SYMBOLIC REASONING OVER NATURAL LANGUAGE

by

Nathaniel Weir

A dissertation submitted to The Johns Hopkins University

in conformity with the requirements for the degree of Doctor of Philosophy.

Baltimore, Maryland

October 2024

Abstract

Much of classical artificial intelligence in the tradition of McCarthy (1959) pursued symbolic theorem proving over carefully hand-crafted ontologies of facts and rules. Due to challenges in scalability and expressivity, such reasoning has fallen out of favor in light of data-driven large language models (LLMs), which capture vast amounts of knowledge and reasoning power by learning patterns over natural language (NL) text corpora. Yet, aspects of the symbolic theorem proving paradigm – namely, mechanistic systematicity, groundedness, and logical underpinnings – are lost in most applications of LLMs, leading to common issues such as hallucinations. This thesis revisits theorem proving in the context of recent advances in LLM reasoning and knowledge retrieval, aiming to leverage the strengths of both the neural and symbolic traditions while addressing their respective pitfalls. I develop a neuro-symbolic entailment engine that reasons over natural language, building on the skeleton of a symbolic prover but leveraging neural models to reason flexibly and dynamically. The entailment engine is designed to systematically determine whether the natural language (NL) hypothesis is compositionally entailed by pieces of evidence provided in large and

readily available text corpora. Through its symbolic search, the engine constructs complex and interpretable entailment trees explaining how complex inferences follow from recursive multi-hop granular inferences, leveraging recent advances in information retrieval and modular reasoning with language models.

While "symbol pushing" engines are semi-guaranteed to arrive at proofs if they exist, the neural variant that I explore in this thesis does not have such a guarantee. NL is infinitely expressive, and by creating an NL proof search, we introduce numerous challenges regarding the *kinds* of inferences and search branches that are most viable to reason over for a given problem. Moreover, by using neural LLMs to generate inferences, we face challenges in getting them to reason robustly. This thesis explores ways to tackle this large and intractable search problem: using a noisy recent tool for reasoning (LMs) for mechanistic backward chaining proof search.

Thesis Committee

Primary Reader:

Benjamin Van Durme

Associate Professor

Department of Computer Science

Johns Hopkins University

Secondary Readers:

Peter Clark

Senior Research Director

Allen Institute for Artificial Intelligence

Daniel Khashabi

Assistant Professor

Department of Computer Science

Johns Hopkins University

Acknowledgments

Thank you to my advisor, Benjamin Van Durme, for his continuing support and unwavering belief in my abilities as a researcher. I am extremely lucky to have had him as a mentor and cheerleader during my challenging PhD journey. Ben gave me confidence in my intuitions and helped me to develop a sense of what makes high-quality AI research. He has taught me to approach my career with healthy doses of optimism, pragmatism, skepticism, and humility. I am very grateful for his guidance.

Thank you as well to my other committee members, Daniel Khashabi and Peter Clark, both of whom have been fantastic mentors. Daniel provided me with valuable insight, perspective, and wisdom as I solidified my research directions in the later stages of my PhD I am eternally grateful to Peter for taking me under his wing as a collaborator and research intern. With little reason to be so generous with his time, Peter has engaged with, supported, and challenged my research ideas, becoming instrumental in inspiring my thesis work. Thank you to the various other mentors and collaborators I have had during my internships at The Allen Institute for AI and Microsoft, including Harsh Jhamtani, Marc-Alexandre Côté, Eric Yuan, Harm Van

Seijen, Bhavana Dalvi Mishra, Oyvind Tafjord, and Kyle Richardson.

I am indebted to my many collaborators on the projects that shaped my thesis work and the course of my PhD. I am grateful to Adam Poliak, who grabbed me by the bootstraps and taught me how to perform and communicate effective NLP research in my first year of graduate school. He and other senior labmates and program members, including Nils Holzenberger, Patrick Xia, Huda Khayrallah, Elliot Schumacher, and Elias Stengel-Eskin were valuable sources of wisdom and technical expertise as I navigated the early stages of my PhD. I am grateful to other co-authors and labmates, including Kate Sanders, Orion Weller, Marc Marone, Joao Sedoc, Shreya Sharma, Zhengping Jiang, Jiefu Ou, Jack Zhang, Anton Belyy, Andrew Blair-Stanek, Aleem Khan, Yunmo Chen, Seth Ebner, and Guanghui Qin. Thank you especially to Kate and Shreya for the long hours spent annotating hard entailment items for the RDTE project, to Orion for his 24-hour retrieval server support, and Marc for serving as a sounding board for ideas and implementations. I am grateful to the many other labmates and collaborators who have provided me with valuable feedback and support over the years.

I was very lucky to have the opportunity to study at Johns Hopkins University and be a part of the large community within the Center for Language and Speech Processing. From my first visits to the campus as a prospective student, I was struck by the sense of community and collaboration throughout the center, and it did not disappoint during my time there. Pursuing a PhD is a challenging and often isolating experience—especially during a pandemic— but I was fortunate to have a

supportive and welcoming community of fellow students with whom I could share both my suffering and my successes. The members of my cohort– Rachel, Carlos, Keith, Alexandra, Liz, Isabel, Chris and Karl– became a second family to me overnight, and I cannot imagine having done graduate school without them. Likewise, to the many other friends made in the CLSP community—Neha, Drew, Aleem, Marc, and others—thank you for your friendship and support. Thank you as well to the members of the Baltimore ultimate community who were so welcoming to me, and to my physical therapists, Gina and Kendra, for helping my ACL rehabilitation after the Seattle ultimate community turned out to be slightly less welcoming.

I would not have gotten so excited about NLP and AI research to pursue a PhD without the guidance and mentorship of my undergraduate mentors and advisors at Brown University. Profs Ugur Cetintemel, Carsten Binning, and Ellie Pavlick exposed me to exciting topics and research opportunities and gave me a taste of the kinds of vibrant discussion and collaboration that I would find in graduate school. My peers in the Brown CS department, especially Ajie, Nick, and the members of the Brown LUNAR Lab helped shape my early research interests and showed me what a supportive research community could look like.

Thank you to my parents, Laurel and Michael, and my sister, Rebecca, for your wisdom and love. Thank you to my grandfather, Ken, for giving me your love of both math and dogs. Thank you to Abbey, who has helped me fight through the dark times and celebrate the bright moments, co-parented our pandemic puppy, edited all

my paper figures, and stole all my grad school friends. Finally, I am grateful to my dog and most frequent office companion, Penny, who has been a constant source of stability, joy, snuggles, and inspiration for my presentation slides.

Dedication

To my mother and father.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Since the early days of artificial intelligence (AI) and natural language processing (NLP) research, the field has pursued creating systems for general purpose, robust reasoning that emulates the cognitive abilities of humans. This thesis will focus on the problem of reasoning over natural language text—a task that is central to the broader goal of creating AI systems that can understand, generate, and perform complex decision making based on natural language input. The ability to reason over natural language text is a key component of many applications of AI, including question answering, information retrieval, and dialogue systems.

Many early "good, old-fashioned AI" reasoning systems were based in a tradition of automated theorem proving. A system would be given a set of axioms and rules of inference, and using logical deduction, it would attempt to derive new facts from these axioms using logical deduction in order to solve problems. These systems were often based on formal logic, and were able to perform reasoning in a systematic, mechanistic and interpretable manner. The core inferential building block of many symbolic theorem provers is the IF/THEN rule, or Horn clause: a statement of the form "If X were true, then we would have reason to believe Y." An example Horn clause from a famous early symbolic reasoning system called MYCIN (Shortliffe and Buchanan, 1975) is shown in Figure 1.1.

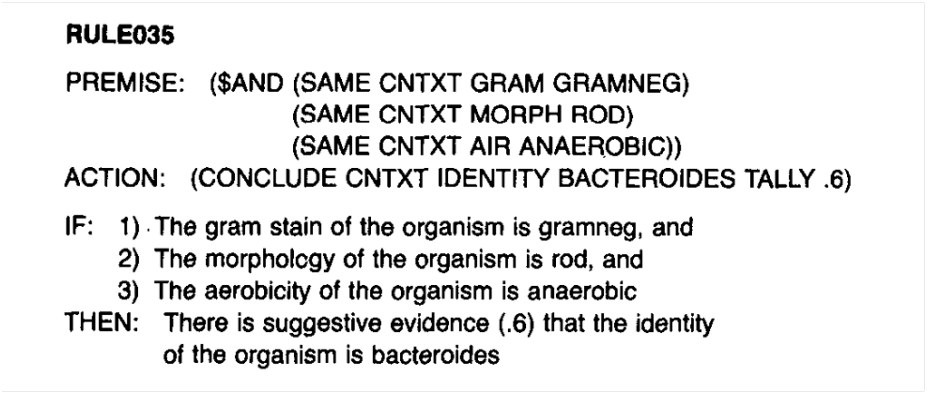


Figure 1.1: Example Horn rule from the MYCIN system. The rule is shown in both its literal form written in LISP and its English translation.

MYCIN was a consultation tool for doctors to make efficient and accurate diagnoses and treatment plans for infectious diseases. It operated by asking the doctor a series of questions about the patient's symptoms and medical history, and then use a set of Horn rules to infer the most likely diagnosis and treatment plan. MYCIN is considered an original *expert system*, an "AI program designed (a) to provide expert-level solutions to complex problems, (b) to be understandable, and (c) to be flexible enough to accommodate new knowledge easily" (Buchanan and Shortliffe, 1984). These qualities are each important in the context of applied AI in scientific domains, and reflect the role of an expert system as an interface between a nontechnical domain expert (e.g. a doctor who does not know programming languages) and the nuts and bolts of an AI inference engine. Critically for the medical domain, MYCIN could provide to its doctor clientele justifications for its conclusions in terms of the rules it used to make the inference. Scott et al. (1977) listed the following key types of explanations producible by their designed MYCIN system:

- how it made a certain decision
- what decision it made about some subproblem
- why it failed to make a certain decision
- why it did not require a certain piece of information
- how it will find out a certain piece of information (while the consultation is in progress)
- what the system is currently doing (while the consultation is in progress)
- how it used a piece of information
- why it did not use a certain piece of information
- why it required a certain piece of information

Figure 1.2: Types of explanation provided by the MYCIN system.

Backward chaining systems like MYCIN inherently support these explanations. They derive hypotheses by explicitly chaining information from their knowledge base. This process allows MYCIN to trace its reasoning, providing a clear rationale for each conclusion by design that can be readily communicated to a human user.

Yet, famously, these systems were brittle and required extensive engineering to encode domain knowledge in a way that was amenable to logical deduction. The "knowledge acquisition bottleneck" plagued AI practicioners and was a major obstacle to the widespread adoption of these systems in real-world applications. MYCIN, for example, required a team of medical experts and AI researchers to encode the rules that it used to make inferences, and the system was only able to reason about the specific domain of bacterial infections for which it was designed. Moreover, it applied only rudimentary NLP techniques to process user queries, allowing for only a limited range of question types and specifications.

Recent advances in deep learning and generative large language models (LMs) have

shown that data-driven neural networks can be used to mitigate, or at least bypass, some of these limitations of traditional symbolic reasoning systems. Neural networks are capable of memorizing vast amounts of knowledge from large-scale text corpora and can generalize to new tasks and domains with minimal engineering effort. These do away with notions of FOL-based knowledge representation, axiomatic inference or bridging the syntax-semantics divide using lexical relations. Instead, they rely upon ever-growing datasets and neural architectures grounded not to theory, but to the principles of 'more data, more parameters, more compute.' Recent generations of LMs approach what might be considered forms of general artificial intelligence; they are able to perform a wide range of natural language understanding tasks with human-level performance on many benchmarks (Hendrycks et al., 2021), including common exams such as the US Medical Licensing Examination (USMLE) (Brin et al., 2023). Yet, these forms of AI are often criticized for their lack of interpretability, and their tendency to rely on superficial patterns in the data to reason in ways that are not acceptable to humans. In a recent slew of instances with real-world implications, they have shown to make up information (the so-called 'hallucination problem'). For example, a doctor recently used an LM to answer a question and found that it cited a journal article in his area of expertise that had been totally made up by the model (Lemoult, 2023). Recent work has found that, even when provided with underlying support documents, state-of-the-art LLMs such as GPT-4 (OpenAI, 2023) can generate medical statements not supported by any medical references at a 30% rate (Wu et al., 2024). Particularly

in domains such as law and medicine for which precise and interpretable complex reasoning is required, this thesis argues that LMs must serve a less all-encompassing role in reasoning systems, even if using them end-to-end is convenient and high-performing.

In light of shortcomings identified in both symbolic and neural reasoning systems, there is a growing interest in developing hybrid neuro-symbolic systems that combine the strengths of both approaches. Some have argued (Marcus and Davis (2019) most prominently) that if the NLP community is to continue to pursue the use of NLMs in cognitive systems that have interpretable structured reasoning, there is a need to revive the desirable qualities of GOFAI expert systems in a way that does not necessarily rely upon NLMs to perform forms of reasoning end-to-end that we are skeptical they can achieve.

1.2 Overview

In this thesis, I reconsider the role of neural language models in artificial cognitive reasoning systems. Rather than relying upon them to perform logical reasoning end-to-end, we consider them as modular components in a hybrid reasoning algorithm reminiscent of classical expert systems that rely upon proof search and hypothesis grounding to knowledge. Specifically, I separate the logic-based symbolic reasoning, for which we rely upon a theorem prover, and knowledge representation and generation components, for which we rely upon NLMs. I pursue a framework for structured

reasoning that performs theorem proving over natural language statements, using NLMs to dynamically suggest large arrays of diverse potential inference steps to be considered by separate critic models. NLMs thus serve as *proposal functions* in a backward chaining search for a structured, semi-symbolic proof over natural language statements, reducing or eradicating the need for engineer-curated knowledge bases of facts and well-defined, brittle rules. In this way, I leverage the generalization capabilities of NLMs to counterbalance the brittleness of symbol-manipulators.

The underlying logical mechanism that I rely on is textual entailment: whether one statement can be inferred from others. This is a fundamental task in NLP that has been used in a wide range of applications, though has ony intermittently been considered in the context of general-purpose problem solving. I build upon recent work that has used NLMs to generate *entailment trees*, a structured representation of a model's reasoning that shows granularly how a natural language hypothesis can be compositionally inferred from a series of simpler statements grounded to a knowledge base. In this way, I develop a system that has the GOFAI-like properties of interpretability, modularity, groundedness, and systematicity and provides the sorts of explanations that systems like MYCIN were designed to readily provide. The ultimate goal of this thesis is a general-purpose *entailment engine* that determines whether some corpus of human-verifiable knowledge compositionally entails a given hypothesis, and if so, generate a structured proof in the form of an entailment tree (Dalvi et al., 2021). An example tree is shown in Figure 1.3.

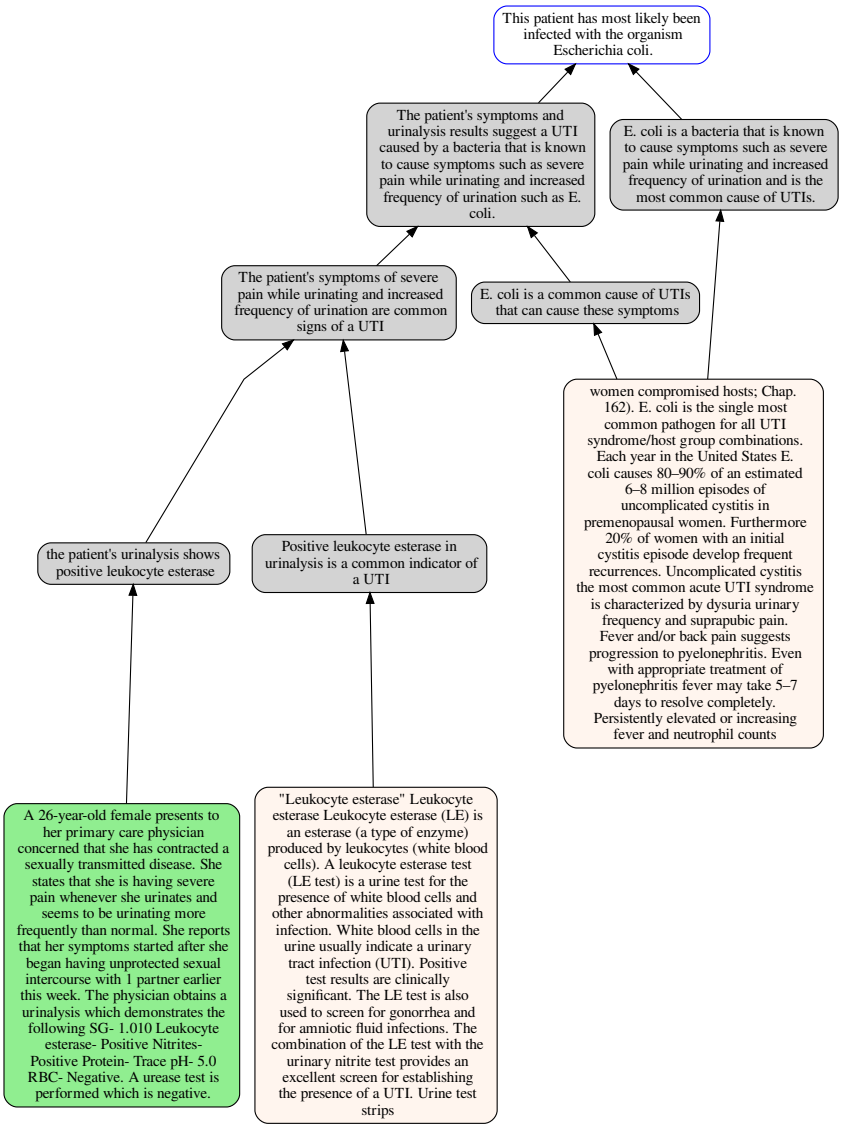


Figure 1.3: Example entailment tree generated by TreeWise, an entailment engine described in this thesis. The tree shows how the hypothesis about a clinical diagnosis can be compositionally inferred from a series of simpler statements grounded in external knowledge documents and the context of the case (green).

Parallel to progress on general-purpose LLM-based reasoners, the NLP field has seen the rise of techniques for *retrieval augmented* generation (RAG; Guu et al. (2020),

Lewis et al. (2020)) and reasoning techniques: methods that "hook up" LLMs to search indices of large text corpora– a resource made increasingly abundant by the rise of the internet– to generate more grounded, interpretable, and accurate text. Compared with a traditional expert system that required immense effort to engineer hand-crafted logic statements, modern retrieval-augmented LLMs can dynamically reason over regular human-written NL text in entire documents of encyclopaedic prose. While the developers of MYCIN had to consult with medical experts to encode the rules that the system used to make inferences, a modern retrieval-augmented LLM can be provided with a corpus of medical textbooks, journal articles, and other authoritative texts to generate inferences about medical conditions and treatments. This thesis utilizes these techniques for systematic reasoning over a large corpus of human-verifiable knowledge, generating structured proofs of entailment against natural language documents.

1.3 Roadmap and Contributions

In Chapter 2, I review existing work on symbolic, neural, and neuro-symbolic approaches to quasi-formal reasoning over text, with a focus on methods that utilize language models for generation of novel textual content as inference steps. I also review some of my own precursor work on extracting and reasoning over text from language models. I also provide an overview of textual entailment in NLP research and the role it has played in evaluating and enhancing reasoning systems.

Chapter 3 then gives an overview of the envisioned framework for an LM-based entailment engine that performs automated theorem proving over natural language statements. I describe the architecture of the system, the role of LMs in a symbolic reasoning-inspired search procedure, and the process of development and evaluation of the first neuro-symbolic compositional entailment engine, NELLIE.

In Chapter 4 I then focus on the core task underlying the entailment engine: recognizing whether a given decomposition of a hypothesis into a series of premises is deductively valid. Collecting realistic and internally consistent data for this task is challenging but is critical to the development of high-quality entailment engines. I describe our pursuit of a theoretically grounded approach to annotating decompositional entailment examples. The contributions of this chapter is a new protocol for collecting entailment data, a new high-quality dataset of decompositional entailment examples, and an evaluation of recent entailment models on this dataset that shows considerable room for improvement.

In Chapter 5, I describe efforts to address some core limitations of the initial NELLIE system, namely the reliance on a clean underlying knowledge source and on in-domain training data. I describe a new entailment engine, TREEWISE, that uses a more flexible and generalizable approach to entailment tree generation that can be readily applied to noisier, longer source texts in different domains of question answering. I evaluate this new engine in a scenario that targets not just correct question answering, but also evaluates the quality of the generated entailment trees.

In Chapter 6, I switch focus to the challenge of populating explicit, generalizable knowledge to improve entailment-based reasoning in a given domain. I explore an application of entailment engines in the automatic construction of natural language *microtheories*, small-scale knowledge bases meant to capture the core concepts and relationships in a given domain. The contributions of this chapter are a question-driven method for populating a microtheory for a question answering dataset and an evaluation of the impact of the resulting microtheories on the performance of entailment engine in the same domain.

Finally, in Chapter 7, I summarize the contributions of this thesis, consider the broader role of entailment-based reasoning in modern NLP, and discuss the limitations of the work, and suggest directions for future research in the field of neuro-symbolic reasoning over text.

1.4 Publications

Portions of this dissertation have been published and preprinted in the following papers:

- (Chapter 3) Nathaniel Weir, Peter Clark, and Benjamin Van Durme (2024a). "NELLIE: A Neuro-Symbolic Inference Engine for Grounded, Compositional, and Explainable Reasoning". *Proceedings of IJCAI 2024*. URL: https://api.semanticscholar.org/CorpusID:258762900

- (Chapters 4 and 5) Nathaniel Weir, Kate Sanders, Orion Weller, Shreya Sharma, Dongwei Jiang, Zhengping Jiang, Bhavana Dalvi Mishra, Oyvind Tafjord, Peter Jansen, Peter Clark, et al. (2024b). "Enhancing systematic decompositional natural language inference using informal logic". *Proceedings of EMNLP 2024*

During my PhD, I have been fortunate to co-author a series of other publications that are relevant to and inform the work that ultimately makes up this thesis. These works can be organized loosely into the following areas:

- **Knowledge Grounding and Attribution in Dialogue and Question Answering**
  - Nathaniel Weir, Ryan Thomas, Randolph D'Amore, Kellie Hill, Benjamin Van Durme, and Harsh Jhamtani (2024). "Ontologically faithful generation of non-player character dialogues". *Proceedings of EMNLP*
  - Orion Weller, Aleem Khan, Nathaniel Weir, Dawn Lawrie, and Benjamin Van Durme (2024a). "Defending Against Disinformation Attacks in Open-Domain Question Answering". *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*. Edited by Yvette Graham and Matthew Purver. St. Julian's, Malta: Association for Computational Linguistics, pages 402–417. URL: https://aclanthology.org/2024.eacl-short.35
  - Orion Weller, Marc Marone, Nathaniel Weir, Dawn Lawrie, Daniel

Khashabi, and Benjamin Van Durme (2024b). ""According to . . . ": Prompting Language Models Improves Quoting from Pre-Training Data". *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*. Edited by Yvette Graham and Matthew Purver. St. Julian's, Malta: Association for Computational Linguistics, pages 2288–2301. URL: https://aclanthology.org/2024.eacl-long.140

- **Guiding LM Generation with Systems of Symbolic Knowledge**

  – Nathaniel Weir, João Sedoc, and Benjamin Van Durme (2020). "COD3S: Diverse Generation with Discrete Semantic Signatures". *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. edited by Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu. Online: Association for Computational Linguistics, pages 5199–5211. DOI: 10.18653/v1/2020.emnlp-main.421. URL: https://aclanthology.org/2020.emnlp-main.421

  – Jiefu* Ou, Nathaniel Weir*, Anton* Belyy, Felix Yu, and Benjamin Van Durme (2021). "InFillmore: Frame-Guided Language Generation with Bidirectional Context". *Proceedings of *SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*. Edited by Lun-Wei Ku, Vivi Nastase, and Ivan Vulić. Online: Association for Computational

Linguistics, pages 129–142. DOI: 10.18653/v1/2021.starsem-1.12. URL: https://aclanthology.org/2021.starsem-1.12

– Nathaniel Weir, Xingdi Yuan, Marc-Alexandre Côté, Matthew Hausknecht, Romain Laroche, Ida Momennejad, Harm Van Seijen, and Benjamin Van Durme (2023). "One-Shot Learning from a Demonstration with Hierarchical Latent Language". *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*. AAMAS '23. London, United Kingdom: International Foundation for Autonomous Agents and Multiagent Systems, 2388–2390. ISBN: 9781450394321

- **Multimodal Neuro-Symbolic Hypothesis Proving**

  – Kate Sanders, Nathaniel Weir, and Benjamin Van Durme (2024a). "TV-TREES: Multimodal Entailment Trees for Neuro-Symbolic Video Reasoning". *arXiv preprint arXiv:2402.19467*

Chapter 2

Background

2.1 Symbolic Reasoning and Expert Systems

2.1.1 Logical Foundations of Reasoning

A massive body of work in applied artificial intelligence has pertained to the design of systems for logical inference over knowledge bases containing facts and rules expressed in some flavor of propositional or first-order (predicate) formalism (Russell and Norvig, 2010). This tradition has origins in Newell and Simon's Logic Theorist (Newell and Simon, 1956) and McCarthy's Advice Taker (McCarthy, 1959), two systems proposed to apply logical operations to manipulate provided symbolic axioms and simulate the human capacity to solve complex problems. The Logic Theorist was applied to prove mathematical theorems in *Principia Mathematica* using four methods for symbolic manipulation: substitution, modus ponens, forward chaining, and backward chaining over a series of very simple logical axioms, e.g., proving the statement $(p \implies \neg p) \implies \neg p$ using axioms such as $p \implies p \vee q$. McCarthy (1959) took this idea a step further, postulating that because the same symbolic manipulation-based search could be performed for any vocabulary of symbols, humans could assign meaning to the symbols in order to solve real-world problems. For example, by codifying rules of common sense into the same sort of logical representation, e.g., $\text{did}(\text{go}(x, y, z)) \implies \text{at}(I, y)$ we can convey to a reasoner what it means to everyday reasoning tasks such traveling locations. McCarthy's Advice Taker could thus hypothetically reason about anything, provided human practitioners tell it the

right information about the world:

> The main advantages we expect the advice taker to have is that its behavior will be improvable merely by making statements to it, telling it about its symbolic environment and what is wanted from it. To make these statements will require little if any knowledge of the program or the previous knowledge of the advice taker. One will be able to assume that the advice taker will have available to it a fairly wide class of immediate logical consequences of anything it is told and its previous knowledge. This property is expected to have much in common with what makes us describe certain humans as having common sense.

Kautz (2022) notes that this key philosophical intuition–that common sense knowledge about location, movement, and many more human-centric systems of knowledge could all be codified interchangeably with pure logical concepts– has roots in Aristotle's *Prior Analytics*, which reflected the insight that reasoning can be performed over the syntactic form of statements without having to understand exactly what the semantics of each statement mean. McCarthy thus proposed artifical theorem proving as a fundamental inferential framework of general-purpose AI. The essential recipe of the framework comprises a (1) *theory* of facts accepted as true and rules for applying and manipulating facts in order to derive new conclusions, (2) a *query* to the theory that might be logically deducible using the facts and rules; the reasoner produces (3) a sequential proof deducing the query if it exists. Figure 2.1 illustrates an example McCarthy (1959) provided of a theory and proof for how the advice taker might take a car to the airport.

Theory:
Facts:

1. at(I, desk)
2. at(desk, home)
3. at(car, home)
4. at(home, county)
5. at(airport, county)
6. walkable(home)
7. drivable(county)

Rules:

1. at(x,y), at(y,z) → at(x,z)
2. walkable(x), at(y,x), at(z,x), at(I,y) → can(go(y,z,walking))
3. drivable(x), at(y,x), at(z,x), at(car,y), at(I,car) → can(go(y,z,driving))
4. did(go(x,y,z)) → at(I,y)
5. (x → can(y)), (did(y) → z) → canachult(x,y,z)
6. canachult(x,y,z), canachult(z,u,v) → canachult(x,prog(y,u),v)
7. x, canachult(x,prog(y,z),w), want(w) → do(y)

(a) Theory: Facts and Rules

Query: want(at(I, airport))
Proof:

1. at(I,desk) → can(go(desk,car,walking))
2. at(I,car) → can(go(home,airport,driving))
3. did(go(desk,car,walking)) → at(I,car)
4. did(go(home,airport,driving)) → at(I,airport)
5. canachult(at(I,desk), go(desk,car,walking), at(I,car))
6. canachult(at(I,car), go(home,airport,driving), at(I,airport))
7. canachult(at(I,desk), prog(go(desk,car,walking), go(home,airport,driving)) → at(I,airport))
8. do(go(desk,car,walking))

(b) Query and Proof

Figure 2.1: McCarthy's Logic for the Advice Taker to Reach the Airport. This framework reflects the general recipe for symbolic theorem proving: a query is deductively proved against a theory comprised of facts and inference rules.

2.1.2 Prolog and Backward Chaining

Since the Advice Taker, numerous logic programming paradigms for automated theorem proving have been established. A popular declarative logic programming language is Prolog, which I will use in this thesis for logically proving queries against a knowledge base. Prolog uses an implementation of the backward chaining proving

algorithm (Hewitt, 1969), in which inference axioms are expressed in the form of rule/body Horn clauses.[1] Prolog reflects the principle of production systems (Post, 1943) that "any rule can fire at any time": each Prolog program enumerates a series of rules that each has the potential to be applied to a query, and the Prolog interpreter will apply rules in a depth-first search whenever the head (the "then" part) of a rule unifies with a given goal or subgoal. Barring being instructed otherwise, the interpreter will try all rules that could possibly apply to a given goal, which can be inefficient in the case of a large number of rules (a scaling issue addressed by a paradigm like answer set programming (ASP; Brewka et al. (2011))) but is effectively guaranteed, with sufficient time budget, to find a proof if one exists in the space of inferences that can be made from the program. Queries to a Prolog program consisting of facts and rules are unified against the program's statements, and an iterative depth-first search is used to find a set of variable substitutions that will prove the query deductively. As as toy example (borrowed from Rocktäschel and Riedel (2017)), consider the following knowledge base:

```
fatherOf(Bart, Homer).
fatherOf(Homer, Abe).
grandfatherOf(X, Y) :- fatherOf(X, Z), fatherOf(Z, Y).
```

Given a goal query such as `?-grandfatherOf(Bart, Abe)`, the backward chaining algorithm first searches for a ground rule with no body, or fact, that can 'unify,' i.e., be set as equal via variable substitution, with the goal term. In the absence of such

[1] It is worth noting that constraining oneself to reasoning over Horn clauses facilitates tractable inference algorithms but can be expressively limited and at times inefficient compared to other options. Other forms of logic programming, such as answer set programming, can be less constraining but are less relevant to the kind of reasoning explored in this thesis.

a fact, it searches instead for a unifiable-head rule with a populated body, whose terms become new subgoals to be proved. In the example, `grandfatherOf(Bart, Abe)` unifies with `grandfatherOf(X, Y)`, and so, upon substitution, the terms of the conjunction `fatherOf(Bart, Z), fatherOf(Z, Abe)` become new subgoals to be proved. Unification with facts `fatherOf(Bart, Homer)` and `fatherOf(Homer, Abe)` prove the subgoals, and so the top-level goal is also proved. If there are multiple ways to prove a goal (e.g. via either `fatherOf(X, Z)`, `fatherOf(Z, Y)` or `fatherOf(X, Z)`, `uncleOf(Z, Y)` through a different rule), Prolog will return all possible proofs, and if there is no way to prove a goal, Prolog will return `false`. Extensions of Prolog include ProbLog (De Raedt et al., 2007), which allows for rules to have associated probabilities that can be used to compute the overall probability of a query being true across all possible proofs.

2.1.3 Expert Systems

The backward chaining theorem proving paradigm gave rise to an entire industry of so-called "expert systems" (Welbank, 1983; Metaxiotis et al., 2002) designed to perform theorem proving in such a way to reflect the decision heuristics of an expert in a given problem domain. From Welbank (1983):

> An expert system is a program which has a wide base of knowledge in a restricted domain, and uses complex inferential reasoning to perform tasks which a human expert could do.

So-called 'knowledge engineers' work in conjunction with domain experts (generally

in a difficult, time-consuming multi-pass process (Musen and Lei, 1988)) in order to curate a sufficient set of inference rules so as to solve reasoning problems in the target domain. Given the flexible expressiveness of predicate logic, these rules can be arbitrarily complex or extremely simple, depending on the choice of knowledge representation and difficulty of the problem. Moreover, declarative logic programming languages like Prolog and Lisp are well-suited for dynamically revising and extending the knowledge base of an expert system, as the rules reflect distinct nuggets of knowledge that can be added and removed at runtime on the basis of performance analysis or new domain knowledge. From Buchanan and Shortliffe (1984):

> The modularity of rules simplifies the task of updating the knowledge base. Individual rules can be added, deleted, or modified without drastically affecting the overall performance of the system. And because each rule is a coherent chunk of knowledge, it is a convenient unit for explanation purposes. For example, to explain why the system is asking a question during the consultation, a first approximation is simply to display the rule currently under consideration.

It is important to distinguish between an expert system and a cognitive model: while some symbolic systems are designed to reflect theories of how humans represent and reason over symbols in their heads, expert systems are designed to only make decisions that *reflect* the same heuristics as experts but might arrive at them in drastically different ways. Humans, e.g., likely do not perform goal-directed backward chaining. From Scott et al. (1977):

> A consultative production system need not be a psychological model, imitating a human's reasoning process. The important point is that the System and a human expert use the same (or similar) knowledge about

> the domain to arrive at the same answer to a given problem. The system's rules and data base can be viewed as a knowledge base containing the domain specific knowledge of an expert as well as facts about a particular problem. When a rule is used, its actions make changes to the data base which are the system's decisions or deductions. Thus, a rule can be thought of as a plece of judgmental knowledge, using the judgment and knowledge of an expert to make deductions.

2.1.4 Symbolic Reasoners over Natural Language

An open challenge in this area is the bridging between NL text and symbolic propositional representations of facts– that is, being able to parse facts and queries stated in freeform natural language text into a machine-tractable knowledge representation. In our running example, imagine if we have the NL evidence, `bart loves his dad, homer` instead of the well-formed propositional fact `fatherOf(Bart, Homer)`. Handling such linguistic ambiguity facilitates the use of natural language data for rule-based machine reasoning rather than having to rely wholly upon expert-curated structured knowledge bases. This is a common challenge faced in many/most NLP tasks pre-neural networks. A common approach to rule-based reasoning over NL text is to perform semantic parsing into a logical form (Moldovan et al., 2003; Blackburn and Bos, 2005; Clark et al., 2014), such as for QA (Green et al., 1961; Zelle and Mooney, 1996) or entailment (Bos and Markert, 2005). If the translation from NL to symbolic representation is accurate, one can leverage fast and scalable solutions for discrete symbolic inference. Examples include theorem provers like Vampire (Riazanov and Voronkov, 2002a), and work in planning as satisfiability testing (Kautz, Selman,

et al., 1992; Kautz and Selman, 1999). However, reliably translating broad-domain NL into an adequately expressive formalism for reasoning is a challenge (Schubert, 2015), as is populating a knowledge base with a sufficient number of inference rules to flexibly support robust reasoning in a given domain, known as the so-called "knowledge acquisition bottleneck," of symbolic which some have argued is an impossible task to overcome (Sowa, 2006). We thus see a core drawback of McCarthy's vision of a general purpose reasoner "improvable merely by making statements to it": figuring out all the statements necessary to make quickly becomes intractable for complex tasks like language understanding. A substantial body of literature has explored ways to tackle the acquisition bottleneck and provide the common sense reasoner its common sense (Van Durme, 2009). Some work compiles text corpora into a noisy and incomplete KB (Niklaus et al., 2018) whose relations, generally corresponding to syntactic patterns, must be interfaced against directly by the inference engine.

2.1.5 Challenges and the AI Winter

Symbolic expert systems achieve certain success on the basis of knowledge bases of hand-written axioms encoding lexical relations and logical operations to handle specific and often fine-grained, domain-specific phenomena. However, they suffer from two common pitfalls of classical expert systems (Musen and Lei, 1988): the knowledge acquisition bottleneck, in which knowledge engineers must laboriously encode relevant domain-specific expertise in a time-intensive and usually multi-phase process, and

the challenge of brittleness, in which an expert system fails to handle a novel case type unforeseen by its developers, possibly due to as simple a mistake as a missing lexical relation in the knowledge base (KB).[2] The problem of brittleness is notably more difficult in the context of reasoning over NL text, which is noisy and has high linguistic variability that lexical axioms struggle to handle comprehensively.

These shortcomings were highlighted in the infamous LightHill report (Lighthill, 1973), which criticized the field of AI for failing to deliver on its promises of general-purpose reasoning systems. Lighthill (1973) argued that while there was viability in certain applications of automation in industrial and scientific settings, e.g., for pattern recognition and signal processing, a large portion of the AI field had focused on an interpretation of AI as a general-purpose reasoning system that aspired to emulate human cognitive abilities. In his words: "an automatic device that mimics a certain range of human functions without seeking in any useful sphere of human activity to replace human being." He argued that these sorts of approaches only found success in very narrow, toy domains in which engineers had successfully abstracted human heuristics into whatever formalism the system was using. To scale up to more general reasoning tasks, he argued, would require a level of knowledge engineering that was infeasible. The report led to a period of reduced funding and interest in AI research, known as the First AI Winter.

[2] Schubert et al. (2010) conclude their discussion with a promise to "enhance both EPILOG itself and its lexical and world knowledge base."

2.2 Neuro-Symbolic Reasoning

With the advent of powerful neural networks, particularly contextual sentence encoders (Devlin et al., 2018; Raffel et al., 2020), the question of handling linguistic variability is considered quite differently. Neural networks are representationally agnostic, flexibly encoding anything that can be expressed as a vector (one-hot or otherwise), thus bypassing the explicit need to settle on a logical formalism such as first-order logic or its various extensions. Data-driven neural networks and, more specifically, neural language models offer a compelling solution to the knowledge acquisition bottleneck that plagued the realistic application of symbolic expert systems. Instead of knowledge engineering, they rely on the *statistical word co-occurrence patterns* contained in increasingly massive swaths of text data to create *distributional representations of meaning* stored continuously in a network's neurons.

Despite their ability to perform extremely highly on many natural language understanding benchmarks (Wang et al., 2018), these neural representations do not have easily interpretable meaning. A subfield of the NLP community centered around the BlackboxNLP workshop[3] has introduced various mechanisms for probing what learned continuous representations of text actually contain and make available for downstream reasoning (Tenney et al., 2019). The capacity to encode continuous vector representations of text that are much more robust to noise and ambiguity has led to

[3](*Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP* 2021)

recent work replacing or augmenting components of symbolic reasoning with neural components.

Nevertheless, neural LM-based approaches to machine reasoning over text continue to face core issues, including a lack of transparency, systematicity, and predictability. LMs operate by modeling some distribution over sequences of language tokens $x \in \mathcal{V}$ for some vocabulary $\mathcal{V}$. The inferences the LM makes are most typically *samples* from a conditional $P(x_i \mid x_1, \ldots, x_{i-1})$ (or $P(y_i \mid x_1, \ldots, x_j, y_1, \ldots, y_{i-1})$ for sequence-to-sequence models that explicitly separate input sequences $\mathbf{x}$ from predictions $\mathbf{y}$). LMs trained for question answering iteratively predict answer tokens conditioned on question previously predicted answer tokens. LMs trained to follow instructions will predict response tokens conditioned on instruction tokens. To sample from this modeled distribution is an inherently noisy, stochastic process,[4] with far fewer structural guarantees of correctness than symbolic reasoning. A symbolic reasoner decides which symbolic predicates are explicitly true given a world model interpretation, an LM decides which symbolic tokens are *likely* to appear in a context. Training the LM to model a distribution of question and answer sequences is not training it to answer questions: it is training it to *guess* which answer words are most likely to appear given a sequence of question words. Bouyamourn (2023) explains how this leads to what is known as the LLM hallucination problem:

> LLMs maximize the conditional probability of the generated strings given the corpus and the prompt. They do not explicitly learn states of the

[4]LM providers advertise this as a feature rather than a bug: increase the 'temperature' (a distribution flattening constant) of your decoding and get more interesting outputs!

> world, do not explicitly learn the meanings of words, and do not explicitly learn the referents of sentences. Each of these types of information is unobserved. As a result, the conditional distributions learned by LLMs can be statistically independent of, or invariant to, their semantic content: that is, of the referents and the truth-conditions of the sentences in the corpus and prompts. So we may have variation in the truth or falsity of a state of the world without variation in the solution to a predictive model that generates the next sentence given a prompt.
>
> Because this semantic information is not contained in the conditional distribution from which an output string is generated, simulating from the learned distribution does not preserve this information, even when it is contained in the corpus. Hence there is no guarantee that LLMs are faithful to the semantic content of their inputs, either. We can think of this as the cause of hallucination in LLMs.

Recent work has explored *retrieval-augmented generation* (Lewis et al., 2020) as a way to enable LLMs to interact with a non-static representation of the world in the form of a retrieval index of documents that can changed or added to in the presence of changing knowledge (e.g., changing the President of the United States Wikipedia page every 4 or 8 years). However, this does not directly address the discrepancy between learning to predict over explicit semantic content and learning to memorize which words appear together to reflect that content.

Marcus (2020), in his prescription for the next decade in AI research, calls for the reintegration of various components of symbolic manipulation missing from the predict-words-in-context neural approach, some of which might address the hallucinatory tendencies of LMs; these include variable binding, logical operations that operate systematically over symbols independent of experience, and a way to explicitly store sophisticated inferences instead of memorizing them:

> Instead of memorizing everything, or interpolating between near neighbors that you might have previously encountered, you draw inferences. Instead of memorizing the fact that Plato and Aristotle and Euripides and each of the other billions of other individuals preceding us were all mortal, you learn general truth, that all humans are mortal, and apply that general truth to specific instances of that category, as needed.

The discrepancies between classical 'symbol-manipulation' and data-driven neural reasoning are a well-trodden yet ongoing subject of debate (Bengio et al., 2019; Marcus, 2020) within both NLP (Hamilton et al., 2022) and other subfields of AI (Renkhoff et al., 2024). Symbolic reasoning is logically sound, domain theory-driven, and nicely facilitates desirable reasoning qualities such as compositionality, symbol grounding, and modus ponens, but suffers from brittleness and extensibility issues. Conversely, neural language models show profound interpolative abilities, picking up on 'vast correlative databases' (Marcus, 2020) to perform well on benchmarks, but they lack systematicity, interpretability, and a working notion of grounded world semantics.

Many in AI have pursued ways to combine the strengths of both paradigms to address the issues inherent to each: symbolic operations for systematic, grounded, and interpretable reasoning and neural networks for flexibility and robustness to noise. The form that this *neuro-symbolic* integration can take varies drastically across NLP and other subfields of AI. Below I will review some categories of approaches to reasoning over language that incorporate both symbolic and neural components, using the typology of neuro-symbolic systems described by Kautz (2022).

2.2.1 Categories of Neuro-Symbolic Reasoning

In his 2020 AAAI keynote address, Kautz (2022) listed a series of categories of ways that symbolic and neural components can interact to create neuro-symbolic systems. Many have noted that these categories, while granular, have somewhat loose boundaries, and it can be difficult to perfectly sort existing methods into their proper buckets. I will describe each and provide recent examples from NLP or other language-centric AI research. Sections are titled with <my descriptor> then (<*Kautz's*>).

2.2.1.1 End-to-End Neural (*symbolic Neuro symbolic*)

This category captures a majority (as Kautz calls it, the "standard operating procedure") of current approaches to applying language models to reasoning tasks: applying them directly end-to-end to the task of interest without any explicit symbolic reasoning component. Most benchmarking in the technical reports of industry-scale LLMs takes this form: hand-annotating a few examples of a task, then applying GPT (Ouyang et al., 2022), Llama (Dubey et al., 2024), Gemini (Gemini Team et al., 2024), or some other LLM to the task, and reporting the results under the assumption that the model is performing the requisite reasoning implicitly. Many have observed that this approach is fallible to dataset leakage: as LLMs are trained on massive amounts of data scraped from the web, they often are able to perform well on tasks by simply memorizing the answers to the questions in the training set (Deng et al., 2024).

In the 2019-2021 era of research, typical end-to-end paradigms entailed finetuning

the NLM on datasets of task examples that purport to exhibit the result of such reasoning types; it makes the assumption that simply by training language models on textual co-occurrences in the right datasets, we can implicitly "teach them to reason" in a specific style (Bhagavatula et al., 2019; Rudinger et al., 2020; Rajagopal et al., 2021; Betz et al., 2021). For example, Bhagavatula et al. (2019) consider the task of *abductive language generation*, in which neural models are tasked with generating an intermediary narrative statement that 'explains' a pair of temporarily ordered 'observation' statements; Rudinger et al. (2020) ask whether models can generate statements that will either strengthen or weaken entailment relations between premises and hypotheses. In both cases, datasets of the desired situational reasoning behaviors are collected, and pre-trained language models are fine-tuned to generate the target statements. Later generations of LM, which can be prompted using a small set of exemplars or zero-shot instructions, can emulate similar types of reasoning without being fine-tuned. This is part of a broader finding that LMs, at a certain size and trained on enough data, can be prompted to generate chains of quasi-systematic reasoning (Nye et al., 2022; Wei et al., 2022).

In tasks such as those in Bhagavatula et al. (2019) and Rudinger et al. (2020) where LMs must *generate novel text* that reflects their reasoning processes, researchers faced challenges as to how to properly sample such text from the model. Prior to advances in instruction tuning that facilitated easy prompting for the kinds of reasoning steps desired, researchers often relied on beam search or nucleus sampling (Holtzman

et al., 2019) to generate text, which often resulted in nonsensical or contradictory text (Rudinger et al., 2020). Aligning the process of LM sequence decoding with the needs of specialized, logically oriented forms of reasoning was highly difficult. In later generations, avoiding hallucinatory generations continues to be a challenge even with the strongest models (Ji et al., 2022).

2.2.1.2 Neural-to-Symbolic Cascade (*Neuro|Symbolic*)

This category uses neural models to convert a problem's non-symbolic inputs into symbolic ones, over which a symbolic solver can then operate. Works that use neural information extraction (IE) models to extract symbolic facts from text documents and save them in a knowledge base for downstream symbolic solvers (Zhou et al., 2022) fall under this category.

More recent work has found success using LLMs to address the semantic parsing of entire problem statements into symbolic analogues (Wong et al., 2023; Lyu et al., 2023; Olausson et al., 2023; Pan et al., 2023; Ye et al., 2023; Gao et al., 2023). Given, e.g. a math word problem, these approaches prompt an LLM to generate a series of symbolic facts and constraints to be fed to a solver (commonly Z3), thus purportedly relying upon the symbolic engine to do the actual "reasoning" and using the LLM only to transduce the NL input into a tractable formalism (in actuality, the LLM will often perform substantial reasoning to arrive at the symbolic representation, e.g., performing steps of the math problem implicitly). This paradigm can suffer from imperfect or

malformed generations by the LLM, though recent approaches have explored ways to generate the programs with more formal verification guarantees of executability (Ryu et al., 2024; Thatikonda et al., 2024).

In general, these approaches have been applied to benchmarks that are either simple math word problems (Cobbe et al., 2021) or are specifically catered toward the kinds of logical operations most amenable to simple first-order-logic problems (Han et al., 2022).

A more complicated approach than directly parsing into symbols is one in which such vectors are treated as *continuous* symbols in place of discrete predicate and argument symbols (Weber et al., 2019; Minervini et al., 2020). These so-called Neural Theorem Provers (NTPs) are a form of inductive logic programming (ILP; Muggleton (1991)) that learns inference rules from observed data.[5] Given a task dataset (e.g., open-domain multi-hop question-answering) and an inventory of *rule templates*, NTPs implicitly learn to encode meaning into each template symbol's embedding.

Backward chaining with NTPs on the surface looks very similar to that of regular Prolog theorem proving, with the added challenge that term symbols to be checked for equality during unification are potentially continuous. This necessitates the notion of a *weak unification* operator, which computes (via cosine similarity) a scalar value estimating the similarity of two continuous symbols. Weak unification scores between two terms result from some monotonic aggregation function over the weak unification scores of their analogous symbols; scores of entire goal proofs are aggregated

[5]See Yang and Deng (2021) for another recent approach to ILP over natural language data without using neural pattern recognition.

similarly over all their unification steps. In our running example, an NTP (specifically, NLProlog (Weber et al., 2019)) would operate over a triplet-ified surface pattern with entities extracted via an off-the-shelf resource such as spaCy: `<X loves his dad, Y>(<bart>, <homer>)`. Weak unification against the subgoal `fatherOf(Bart, Z)` would compute the embeddings of and aggregate the similarities over the pairs (`<X loves his dad, Y>`, `fatherOf`), (`<bart>`, `Bart`), (`<homer>`, `Z`).

2.2.1.3 Symbolic with Neural Subroutines (*Symbolic[Neuro]*)

This category refers to predominantly symbolic reasoners that use a neural component to make certain internal decisions. The colloquial example is AlphaGo (Silver et al., 2016), which follows a symbolic Monte Carlo tree search algorithm operating in a symbolic problem space that uses neural state estimators to inform decision-making. More broadly, there is substantial literature on using neural models to guide or constrain the search space of the symbolic algorithm. Li et al. (2024) give an overview of how LMs and other neural approaches have been used in formal theorem proving, e.g., for premise selection, proofstep generation, and heuristics to guide symbolic proof search. Some use LMs to generate proof steps in mathematical theorem proving (Polu and Sutskever, 2020; Welleck et al., 2022).

In NLP, this category covers methods that use LMs and other neural methods to interface dynamically with text (as opposed to after-the-fact, as in the previous category), which is either user-provided or LM-memorized. Some have treated LMs as

"embedded knowledge bases" that can be dynamically queried to instantiate predicates for a symbolic algorithm (Petroni et al., 2019). Bosselut et al. (2019) introduce a model called COMET (COMmonsEnse Transformers) trained to instantiate a predefined set of predicates linking together a provided NL text span with generated one (e.g. causeOf("the glass broke", ?) → "the glass was dropped"). Various works have used COMET as a subroutine within broader, more-or-less symbolic frameworks, including CLUE (Arabshahi et al., 2021a) (an extension of the NTP, CORGI (Arabshahi et al., 2021b)) for conversational inference, COMET-DynaGen (Bosselut et al., 2021) for commonsense question answering, and CAST (Peng et al., 2022). In a more symbolic search-inspired direction, the Braid system Kalyanpur et al. (2021) also proposes an application of embedded KBs to the story understanding domain; they augment a hand-curated rule-base of axioms with a dynamic predicate-generating T5-GLUCOSE model (Mostafazadeh et al., 2020), creating a Prolog-style backward chaining theorem prover. Their engine performs inferences mainly using a FOL-based semantic formalism, bridging between NL and the logical representation using a semantic parser while converting in the other direction using rule-based language generation. With the exception of Braid,[6] most of these approaches generally do not address the challenge of stochastic sampling from LMs causing noisy inference– much of the time, the final decision by the system is based on an aggregated confidence metric (inference chain X was assigned more likelihood than Y by the LM) rather than grounding a response in

[6]Kalyanpur et al. (2021) do not evaluate their system, so it is unclear whether it works at all.

a directed inference chain grounded to knowledge statements as symbolic reasoners do. The systems that I introduce in Chapter 3 and Chapter 5 will be Symbolic[Neuro] systems that attempt to address this challenge. Other algorithms that use LMs to systematically construct entailment trees from evidential statements as discussed in Section 2.3 also fall in this category.

Another class of architecture using neural subroutines is neural module networks (NMNs; Andreas et al. 2016), where different neural models are chained together dynamically to solve tasks with nonsymbolic inputs. Gupta et al. (2020a) propose a textual QA-oriented NMN for solving tasks requiring text span extraction and comparative and numerical operations (theirs is also a cascading neuro-symbolic system, as its first step is to parse a question into a logical form that invokes the neural modules). Khot et al. (2021) introduce another NMN that decomposes questions into simpler ones answerable by existing models. As LMs have advanced to prompt-based reasoning methods, newer NMN-like frameworks often involve systematically delegating different problem-solving steps to differently-prompted versions of the same model (Khot et al., 2023).

Also in this class of neuro-symbolic reasoning are more recent systematic algorithms resembling symbolic methods wrapped around so-called chain-of-thought reasoning. In these algorithms, the atoms around which the symbolic methods operate are often either NL statements or entire NL reasoning chains. These methods include self-consistency (Wang et al., 2023) (majority voting of answers from multiple chain-of-

thought samples), Tree of Thoughts (Yao et al., 2023a) (a self-pruned BFS across possible chain-of-thought inference steps), and self-evaluation guided beam search (Xie et al., 2023).

2.2.1.4 Neural with Symbolic Subroutines (*Symbolic[Neuro]*)

The inverse of the previous category refers to predominantly neural approaches that invoke symbolic mechanisms dynamically as subroutines and feed the inputs back to the neural reasoner. A popular example is "tool use," whereby LMs are enabled to dynamically invoke external tools like calculators or APIs to solve complex tasks (Qu et al., 2024); the workflow alternates between the LM planning, selecting, or calling tools, and the tools executing and returning information to the LM for future steps. Iteratively prompting a retrieval index and generating new inferences from the LLM constitutes another version of tool use under this category (Trivedi et al., 2023). Poesia et al. (2024) leverage a tool-invoking-like mechanism to improve the deductive validity of LM reasoning through logically constraining "guides," yielding a reasoner that can interchangeably reason with NL and FOL. Another popular framework is ReAct prompting (Yao et al., 2023b), which involves neural models interleaving verbal reasoning and planning steps with execution actions, thus dynamically adapting to additional information and feedback from an external symbolic environment. ReAct style prompting generally relies upon handcrafted exemplars in a domain to convey task-specific details to the model, though more domain-general approaches with less

manual overhead are an open area of research.

2.2.1.5 Symbolic Compiled into Neural (*Neuro: Symbolic* $\rightarrow$ *Neuro*)

This category refers to approaches that use neural models to "compile away" symbolic structure and/or emulate symbolic behavior. The canonical example is Lample and Charton (2020), who trained a neural network to compute function integrals and solve complex differential equations.

One subclass of work falling in this category is work that explores whether LMs trained on symbolic or logically infused data can reason logically or perform better on logical reasoning benchmarks (Kassner et al., 2020; Betz et al., 2021; Traylor et al., 2021; Saeed et al., 2021; Rozen et al., 2021).

Other recent work explores methods of handling NL-like versions of fact/rule theories (i.e., small symbolic reasoning problems cast into NL) without parsing them to another formalism. These commonly evaluate against the RuleTaker dataset introduced by Clark et al. (2021) (also referred to as the ProofWriter dataset). Some work explores using LMs to emulate stepwise (Tafjord et al., 2021; Saha et al., 2020; Kazemi et al., 2023) or end-to-end (Clark et al., 2021; Picco et al., 2021) logical reasoning over small rule bases. These approaches require both user-provided if/then rules to operate, while the approaches considered in Chapter 3 and Chapter 5 require only facts and thus apply to domains in which rules are not available.

2.2.1.6 Structurally Symbolic Neural ($Neuro_{Symbolic}$)

In this category, various approaches use symbolic logic as templates for structures with the architecture of the neural network performing the reasoning, thereby often imposing constraints on the network's behavior. An example of this is Tensor Product Representations (TPRs; Smolensky et al. 2016), which use an alternative to typical neural network neurons based on the tensor product operation that explicitly model variable relations, objects, and bindings. Other architectures include Logic Tensor Networks (Badreddine et al., 2020), which integrate first-order-logic predicates in continuous tensor space using fuzzy logic semantics, Neural Tensor Networks (Socher et al., 2013), which learn vector representations for entries and relations in a KB, and Semantic Probabilistic Layers (Ahmed et al., 2022), which can replace standard neural output layers but enforce logical constraints on the output space.

These sorts of solutions have been applied relatively sparingly to NLP tasks, given their difficulty in implementation (partly due to a lack of widespread adoption in standard modeling packages), high computational complexity, and limited empirical success on complex problems.

2.2.1.7 Categorizing This Thesis

Where does the work in this thesis fit into the broader space of neuro-symbolic research for reasoning over text? Chapter 3 and Chapter 5 describe reasoning algorithms coded in Prolog, a symbolic reasoning language that implements a typical

backward-chaining symbolic search. Yet, the symbols over which the algorithms reason are all NL sentences: they introduce new "words" into the Prolog engine's vocabulary at each step using language models and neural retrieval modules. As the control system or orchestrator of the algorithm is symbolic, I put NELLIE and TREEWISE in the Symbolic[Neural] category. It also incorporates elements of the Neuro: Symbolic $\rightarrow$ Neuro category, as the neural components are used to "compile away" the (soft) logical relationship between the input knowledge and entailed hypothesis symbols. The work in Chapter 6 is also Symbolic[Neuro], as it extracts knowledge statements from an LLM as an "embedded knowledge base" and feeds the statements to a Symbolic[Neuro] reasoner for inference.

2.3 Entailment

The task of recognizing textual entailment (RTE), commonly referred to interchangeably as natural language inference (NLI), refers to the problem of determining whether a natural language hypothesis statement h can be reasonably inferred given a natural language premise p.

The notion is drawn from formal linguistics, with an important distinction pertaining to definitional rigor. In formal semantics, 'entailment,' referred to as "the basic aim of semantics" by Montague et al. (1970), is a technical term strictly defined with respect to world models: "sentence [p] entails sentence [h] if in all models in

which the interpretation of $[p]$ is true, also the interpretation of $[h]$ is true." (Janssen and Zimmermann, 2021). In other words, h must necessarily be true in every possible situation in which p is true. This definition is naturally predicated upon well-formed notions of "model" and "interpretation," which are available in the symbolic languages typically considered in formal semantics but not in natural language processing applications. This strict definition is thus often too rigid for practical NLP tasks, for which it is often impossible and/or impractical to determine exact entailment.

In contrast, NLP practitioners take a much looser, human-centric approach to defining *textual*, as opposed to formal entailment. As stated by Dagan et al. (2005):

> We say that T entails H if the meaning of H can be inferred from the meaning of T, as would typically be interpreted by people. This somewhat informal definition is based on (and assumes) common human understanding of language as well as common background knowledge

Manning (2006) further specifies that 'people' in this definition should refer to "people that are awake, careful, moderately intelligent and informed, and with reasonable document interpretation expertise [...]. These people would use whatever background and real-world knowledge that they usually bring to interpreting texts."

This definition of textual entailment also necessarily introduces a degree of acceptable uncertainty. For example, a person reading the following pair would generally agree that it constitutes entailment:

(P) Penny is walking in Baltimore

(H) Penny is walking in Maryland

Under strict logical entailment, there exists extremely unlikely situations in which (P) holds but (H) does not; for example, "Baltimore" might refer to Baltimore, Ohio instead of Baltimore, Maryland. Yet, readers with a typical level of world knowledge would eschew this interpretation as sufficiently unlikely that it would be impractical to not accept (H) as true. For this distinction, many practitioners prefer the terms "NLI" or "textual inference" to "textual entailment," though this thesis will not treat them particularly differently.[7]

The focus on what humans know and will accept on incomplete evidence arguably reflects a common intuition that many popular NLP user-facing end tasks such as translation, summarization, question answering, and semantic search are all predicated upon, or at least benefit from (see §2.3.2), the ability to perform RTE (MacCartney, 2009). Dagan et al. (2005), in introducing the seminal PASCAL RTE challenge, described it as an "attempt to promote an abstract generic task that captures major semantic inference needs across applications." As many such use cases 'need' to emulate and/or agree with human reasoning processes, it was pertinent to define entailment in such a way to be grounded in empirical judgment (human labels, i.e., an ultimately "know it when I see it" mentality). The implication of this is that NLP researchers avoid committing to definitions of entailment whenever possible, and entailment is quantified through collecting judgments from human annotators using

[7]I will note that for my purposes building systems to determining complex entailments through both generating and discerning valid sub-entailments, RTE best fits the classification task in which a premise and hypothesis to an agent for labeling, while NLI better fits the generative scenario in which an agent seeks to produce the premises that would support a provided hypothesis.

a variety of more-or-less vague instructions rather than formal, abstract theory. As humans do not think as stringently as logical axioms (Sulik et al., 2021; Tan, 2022), they, and accordingly, NLP conceptualizations of entailment, tend to allow "almost certain" inferences as opposed to "absolutely guaranteed," and to allow for certain kinds of common knowledge to be left out of premises. From Manning (2006):

> One way of thinking about whether an hypothesis follows from a text is: if a person asked you for a piece of text that establishes a certain hypothesis, then would showing them the given text satisfy the person. From an operational perspective, this seems just what we want. For instance, think of applications like passage retrieval, question answering, or event extraction. It is not useful to provide any document at all as a justification for something that we know to be true from a database table or knowledge base. On the other hand, it is very reasonable to return the following text in support of the given hypothesis, even though its sufficiency is dependent on knowledge that Paris is in France:
>
> Text: The Mona Lisa, painted by Leonardo da Vinci from 1503-1506, hangs in Paris' Louvre Museum.
>
> Hypothesis: The Mona Lisa is in France. FOLLOWS. (RTE1 ID: 153)

See Clark et al. (2007) for a characterization of the types of lexical, world and background knowledge that the PASCAL challenge assumed for full justification of the entailments.

NLP research has embraced the probabilistic factor as a necessary consideration for many kinds of textual entailment problems of interest to the community, e.g., in common-sense situational reasoning. Works such as Zhang et al. (2017) and Chen et al. (2020) treat entailment as having ordinal and scalar certainty, respectively, as opposed to a binary yes/no or ternary entailment/contradiction/neither nature.

Yet, with the community's acceptance of implicit knowledge and uncertainty comes

the cost of a lack of a shared, codified protocol for determining validity across the many domains for which textual entailment has been considered. Pavlick (2017) notes that "[b]ecause the NLP notion of entailment is intentionally informal and underspecified, it leaves ample room for variation across datasets as what constitutes "common human understanding of language" and "common background knowledge."" Such underspecification can also lead to natural inherent disagreements between annotators when labeling the same NLI pairs, as observed by Pavlick and Kwiatkowski (2019) and Chen et al. (2020). This thesis will not attempt to solve this problem but rather acknowledge its influence on our design of entailment elicitation protocols in Chapter 4, taking into careful consideration the audience of the reasoning traces produced by entailment-based reasoners and the extent to which these traces rely upon common versus esoteric knowledge.

2.3.1 Entailment as a Medium for Intrinsic Evaluation

Unlike in the chapters that follow, entailment on its own has rarely been considered by NLP practitioners to be an end task.[8] Instead, it has often served as a common medium for *evaluating* models for various purposes, many of which fall into 4 (non-exclusive and noncomprehensive) buckets:

[8]In the words of my advisor: "Who cares about entailment as long as my pizza arrives on time?"

2.3.1.1 Probing for whether models can handle specific semantic or linguistic phenomena

The FraCaS Test Suite (Cooper et al., 1996), carefully hand-annotated by linguists to evaluate models on categories such as quantification and negation, introduced the concept of using entailment to diagnose model shortcomings on specific classes of linguistic phenomena. Datasets such as SICK (Marelli et al., 2014), DNC (Poliak et al., 2018), and numerous others have followed suit (White et al., 2017; Richardson et al., 2020; Hossain et al., 2020). A similar line of work has identified common patterns and heuristics that trip up models through systematically designed challenge sets (McCoy et al., 2019; Naik et al., 2018).

2.3.1.2 Probing for certain tasks and reasoning paradigms

Dagan et al. (2005) made the observation (later followed by White et al. (2017) and Poliak et al. (2020)) that many real-world NLP tasks could be recast into entailment examples. They released a series of 8 shared RTE tasks based on other tasks such as reading comprehension, information extraction, and paraphrasing (this push of recasting was later followed by White et al. (2017) and Poliak et al. (2020) using semantic resources). Wang et al. (2018) similarly recast the SQuAD QA dataset (Rajpurkar et al., 2016) and Winograd Schema Challenge (Levesque et al., 2012) into NLI.

While most of the above tasks imposed a general notion of deductive validity, other work has considered more specialized versions of NLI, particularly in the context of situational commonsense reasoning. Bhagavatula et al. (2019) recast the ROCStories story cloze dataset (Mostafazadeh et al., 2016) into a nominally 'abductive NLI' (αNLI) scenario in which models must predict which of two events is more likely to have occurred between a pair of other events. Rudinger et al. (2020) augmented a series of datasets including SNLI (Bowman et al., 2015) into a 'defeasible NLI' (δNLI) task in which models must determine whether an auxiliary 'update' statement would strengthen or weaken a given entailment. Both αNLI and δNLI introduced a generative variant of their classification tasks, though neither generative evaluation was tied to an end task. As mentioned above, Zhang et al. (2017) and Chen et al. (2020) both collected re-annotations of NLI datasets using ordinal and scalar values, thus explicitly modeling the level of certainty that a given hypothesis would occur given the premise.

2.3.1.3 Entailment against complex or heterogeneous kinds of premises

While early NLI tasks generally consider 1-sentence or short-passage premises, numerous works have reconsidered the forms and sources of the premises, e.g. semi-structured tables (Gupta et al., 2020b), non-English languages (Conneau et al., 2018), Wikipedia snippets (Clark et al., 2019), and entire naturally occurring documents (Yin et al., 2021).

Of particular relevance to this thesis is compositional, multi-premise datasets such as MPE (Lai et al., 2017), QASC (Khot et al., 2020), eQASC (Jhamtani and Clark, 2020), and BaRDa (Clark et al., 2023) that ask whether a hypothesis follows from an unordered conjunction of two or more statements. All but MPE were created with an end task in mind (multiple-choice ScienceQA); the premise sets are meant to provide valid justifications for correct answers. However, these datasets allowed substantial freedom to crowdsourced annotators to choose what constitutes entailment. MPE, which is based on simple photo captions as premises, specified that entailment meant "whether or not the scene described by the four captions can also be described by the proposed [hypothesis]." QASC/eQASC specified that " the two facts combine or "chain" together to explain an answer in a sensible way. Or, in some cases, just one of the facts is sufficient to justify the answer. " (Jhamtani and Clark, 2020). In Chapter 4 we will discuss the implications of underspecified instructions for annotating multi-premise decompositions of this sort, as these datasets show to yield weak models for determining entailment validity in practice (specifically, in the context of entailment engine proof search).

2.3.1.4 Entailment for the sake of entailment (leaderboards)

While in the above three categories, entailment has been used as a means towards evaluating specific types of reasoning and phenomena, other datasets have been constructed mainly as a means for models to get better at entailment more generally.

SNLI (Bowman et al., 2015), MNLI (Williams et al., 2018), and ANLI (Nie et al., 2020) were each created primarily for this purpose. As Poliak (2020) describes, researchers focused on these tasks and their very popular leaderboards primarily as a means for competition on a dataset and less as a way to consider the implications of successfully using entailment to evaluate how well models actually perform at real tasks.

2.3.2 Using Entailment for Downstream Tasks

In contrast to the above works, the entailment engines we will consider in this thesis will treat RTE as a mechanism for improving systematic and interpretable reasoning. I am not the first to argue that RTE can serve an explicit role in improving reasoning systems, various NLP researchers have advocated for the effectiveness of RTE models in improving the performance of systems on various downstream tasks such as question answering, relation extraction, machine translation, and summarization (Bentivogli et al., 2017).

Clark et al. (2012) presented a mechanism for QA over single documents via estimating entailment between source and answer sentences. Trivedi et al. (2019) used pre-trained neural NLI models as a mechanism for selecting relevant premises for multi-hop question answering. Other works have found success using NLI as decision verifiers on top of existing systems. Harabagiu and Hickl (2006) explored different ways to incorporate RTE systems to filter or rank answers by an open-domain QA system that used retrieval and mid-2000s NLP techniques. This idea was echoed over

a decade later by Chen et al. (2021), who recast question/answer predictions into NLI pairs using QA2D declarativization (Demszky et al., 2018) to verify that answers are sufficiently supported by context passages.

Many researchers in the late 2010s found that NLI is a useful tool for bootstrapping models trained for another task, showing that neural models pre-trained on corpora that contained large NLI datasets like SNLI and MNLI yield stronger finetuned models on other downstream tasks (Phang et al., 2018; Wang et al., 2019). They found moderate success in such multitask and transfer learning scenarios (Guo et al., 2018a; Guo et al., 2018b), though it is unclear whether NLI data still has such an influence on later generations of models like GPT-3, GPT-4, and Llama.

2.3.3 Models for Entailment

The most popular current approach for modeling entailment is the neural entailment classifier: a pretrained encoder (Devlin et al., 2019) or encoder-decoder (Raffel et al., 2020) model fine-tuned on one or multiple NLI datasets. More specifically, practitioners use a semi-supervised language modeling and/or denoising objective to pre-train these models on very large text corpora. They then fine-tune the model on NLI datasets. In the encoder-decoder case, NLI is modeled as a text-to-text task, mapping the concatenation of the premise and hypothesis to a positive or negative label word (e.g., the word "entailment," "neutral," or "contradiction"). In the encoder case, practitioners add a "classifier head" that is trained to accept the contextual encoding of an input

sequence by the pre-trained encoder and predict a probability distribution over a provided set of labels. This approach is often called a "cross-encoder" model because by accepting the concatenation of the two sequences, the encoder implicitly models interactions between terms and phrases. I will make use of cross-encoder NLI models frequently in Chapter 3 and Chapter 4.

Prior to the 2015 neural network revolution, systems for RTE took very different approaches to modeling the problem. As with many NLP tasks, a common framework was to parse the NL premise and hypothesis into a logical formalism using a semantic parser and then apply logical deduction algorithms to check for entailment. Nutcracker (Bos, 2014) is a representative example of this, using the Boxer system for semantic parsing (Bos, 2015) and the Vampire (Riazanov and Voronkov, 2002a) system for deduction. Nutcracker followed common precedent by using the lexical relations stored in WordNet (Fellbaum, 2010), as well as a small set of hand-crafted axioms derived from NLI development sets. The Epilog system (Schubert et al., 2010) epitomizes a heavier application of hand-crafted axioms to perform NLI and a wide array of common sense inferences. Notably, Bos and Markert (2005) found that their Nutcracker system was ultimately not any stronger than a less robustly defined approach that relies only on shallow semantic analysis (e.g. string similarity) rather than deep analysis via Boxer. Later systems embraced the shortcomings of purely logical inference-based approaches by blending the proof-based approach with shallower statistical methods (Lewis and Steedman, 2013). Others considered ways to

overcome the bottleneck of annotating entailment rules by automatically constructing them (Berant et al., 2011).

2.3.4 Entailment Trees

An important recent development toward a more direct application of textual entailment for downstream tasks and general reasoning is the introduction of entailment tree-based reasoners (Dalvi et al., 2021). Entailment trees are multi-level directed graphs (not technically trees because children can have more than one parent) of natural language statement nodes that show how a hypothesis is *compositionally* entailed by child premises.

Dalvi et al. (2021) introduced EntailmentBank, the first dataset of entailment trees. It was presented as an explanation dataset for science question answering, building upon a long effort by Peter Clark and collaborators to gather and reason systematically over the kinds of knowledge required to achieve strong performance on grade school science tests (Clark et al., 2020). Each tree in the dataset is rooted at a hypothesis that represents the correct answer to a question from the AI2 Reasoning Challenge (Clark et al., 2018) and represents one way to decompose the hypothesis into simpler statements that combine to support accepting the hypothesis as the correct answer. The tasks associated with the released dataset are generative rather than discriminative, focusing on evaluating models' abilities to generate a good argument supporting the correct conclusion. While in typical RTE, the task is to answer **whether** a hypothesis follows

from a premise, EntailmentBank asks models to answer **why** the hypothesis follows in a granular and structured fashion. This spirit is echoed in the later STREET benchmark (Ribeiro et al., 2023). In this way, EntailmentBank is the first dataset for multi-hop, task-oriented, generative entailment, and also one of the first datasets to propose entailment as a means for creating logically sound, directed, and grounded explanations stated primarily in natural language. This was a significant contribution to the growing discourse around developing explanation generation methods and, more broadly, explainable AI systems that leverage powerful but inherently black box neural models (Wiegreffe and Marasovic, 2021; Tan, 2022; Lyu et al., 2024).

EntailmentBank has served as a training resource for approaches to tree generation that build upon the baselines introduced in their paper. These include SCSearch (Bostrom et al., 2022), IRGR (Ribeiro et al., 2022), METGEN (Hong et al., 2022), and NLProofS Yang et al. (2022). It also serves as the basis for training Entailer (Tafjord et al., 2022) and TeachMe (Dalvi Mishra et al., 2022), two QA systems that operate by decomposing hypotheses into entailing conjunctions of premises believed by the model and/or gathered during offline training.

EntailmentBank provided ample research questions surrounding the novel notion of entailment trees in the age of deep learning. The dataset also did not directly address question answering itself– the evaluation was focused on reconstructing trees for only correct answers and not on whether models could simultaneously generate high-quality trees **and** answer multiple-choice questions correctly. In other words,

EntailmentBank poses to models the following scenario:

> Here is a hypothesis and a corpus of text. We know that the corpus entails the hypothesis. Your task is to show how.

What is a different, and perhaps harder, but more realistic scenario, is as follows:

> Here are n conflicting hypotheses and a corpus of text. We don't know whether the corpus entails one or any of them. Your task is to figure out which one(s) is entailed, *and* to show how.

I will consider this different entail question across Chapter 3[9] and Chapter 5.

The dataset also did not directly address the question of explanatory completeness or the sufficiency of entailments contained within each tree ("Our focus here is on generating the derivation (line of reasoning) showing how the evidence leads to the answer, rather than the pragmatics of deciding which parts of that to then show the user. " (Dalvi et al., 2021)), thus following the community's aforementioned precedent of defining entailment vis-a-vis the individual heuristics of annotators. I will consider this open challenge in Chapter 4.

[9]Chapter 3 will consider evaluations where we know that the corpus at least *supports*, if not entails, the correct hypothesis. When we expand the evaluation to include Wikipedia as a corpus in Chapter 5, we will no longer have any guarantee.

Chapter 3

NELLIE: A Neuro-Symbolic Entailment Engine for Grounded, Compositional, and Explainable Reasoning

3.1 Motivation and Overview

Large language models (LLMs) are impressive at question-answering (QA), but it remains challenging to understand how answers systematically follow from authoritative information. This general opacity is a growing impediment to widespread use of LLMs, e.g., in critical applications such as medicine, law, and hiring decisions, where interpretability and trust are paramount. While there has been rapid progress in having LLMs generate systematic explanations for their answers, e.g., Chain-of-Thought (Wei et al., 2022), Entailer (Tafjord et al., 2022), or Maieutic Prompting (Jung et al., 2022), such explanations are not grounded in external facts and may include hallucinations (Ji et al., 2022). Rather, what would be desirable - and what this chapter pursues - is a system that systematically reasons over authoritative text to support answers with human interpretable proof trees grounded in the text while not requiring translation to an entirely symbolic formalism.

Our approach is to revisit the behavior of *expert systems* (Jackson, 1986; Metaxiotis et al., 2002). As described in Section 2.1, expert systems are appealing for their explainable behavior: decisions are made by constructing a well-formed symbolic proof from explicit, formally represented knowledge authored by a knowledge engineer in consultation with a domain expert. However, as expert systems are known to be both expensive and brittle (Musen and Lei, 1988), the AI community has turned to *neuro-symbolic* mechanisms that use large language models to reason over natural language (NL). Reasoning over NL does not require a symbolic formalism and allows the use

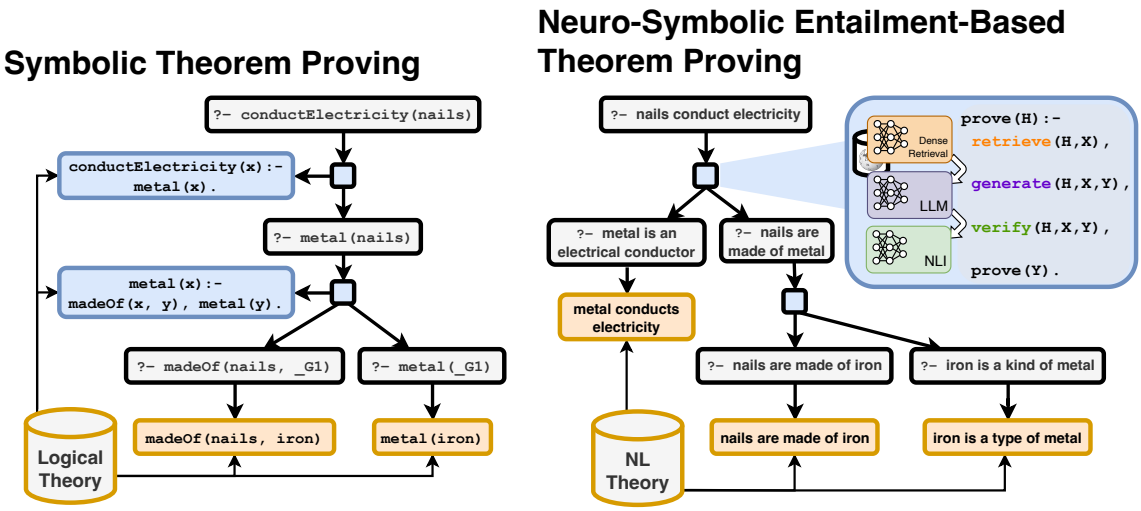


Figure 3.1: Symbolic reasoners **(left)** prove a hypothesis using a search over facts and rules expressed in a symbolic logical formalism. This chapter (and this thesis more broadly) explores a neuro-symbolic variant of the search **(right)** that uses natural language facts and entailment as the logical underpinning. Meta-rules fed to the reasoning engine invoke neural modules for retrieving facts, generating inferences, and verifying decompositions.

of inferentially powerful LLMs, but also loses the systematic, verifiable proofs that expert systems produced. This motivates our pursuit of a new way to jointly reap the benefits of both modern neural methods and traditional reasoning.

We desire the following expert system-inspired desiderata:

1. **Grounding** inferences fully and scalably in a corpus of knowledge from an authoritative human source.

2. **Logical direction**, showing how a given hypothesis (and all intermediate inferences leading up to it) follows as the logical consequence of the underlying knowledge source

3. **Competitive end-to-end QA performance** in a complex domain requiring

diverse forms of reasoning.

As illustrated in Figure 3.2, various eXplainable QA (XQA) methods accomplish 2 of these criteria but not all 3. Some methods, like Chain-of-Thought, generate inference chains or logically structured graphs from an LLM without grounding in verified knowledge. Others, like ExplanationLP (Thayaparan et al., 2021), compose graphs of grounded facts but do not show logical direction. LAMBADA (Kazemi et al., 2023) achieves both direction and grounding but is limited to simple domains with provided NL Horn rules and sets of 1-2 dozen facts. In contrast, we desire a system that handles a larger corpus (10K statements) and doesn't need handcrafted rules.

To achieve all three desiderata, we reuse the general inference framework of an expert system but replace handcrafted rules with a combination of neural language modeling, guided generation, and semiparametric dense retrieval. We demonstrate this in a system called NELLIE, the **Ne**uro-Symbolic **L**arge **L**M **I**nference **E**ngine. NELLIE leverages LLMs as *proposal functions* in the search of proof trees showing how a conclusion follows via entailment from an external corpus of NL statements. The "symbols" that our neuro-symbolic engine reasons over are free-form NL sentences. Rather than require a knowledge engineer to carefully write hundreds of inference rules as in the classical setting, NELLIE employs an LLM as a *dynamic rule generator* (DRG; (Kalyanpur et al., 2021)) to *generate* (rather than retrieve) candidate rules that decompose a hypothesis into subqueries that, if themselves proved recursively, will prove that hypothesis via composition (Figure 3.6). In this way, NELLIE can answer

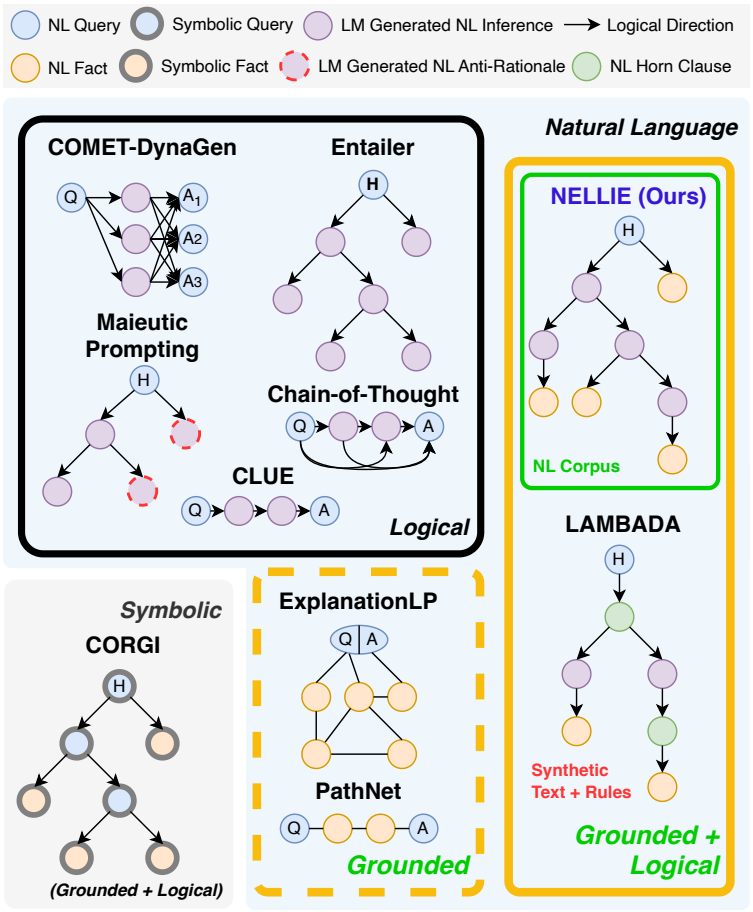


Figure 3.2: Comparison of approaches to neural XQA. Each approach leads to NL graphs in support of a query. Our proposal is to produce logically directed explanations containing model-generated intermediate steps while grounding a tree in verified facts without relying on handwritten horn clauses.

the question posed by the classical expert system: "Does this statement systematically follow from my knowledge, or not?".

NELLIE is built upon a backward chaining symbolic theorem prover written in Prolog, implemented using a few *meta-rules* specifying how inference should proceed. These include checking for entailment of a hypothesis against a retrieved fact or decomposing it into a conjunction of subqueries to recursively prove. To treat NL

sentences as if they were symbols in a purely symbolic search algorithm, we use a natural language inference-based form of *weak unification* (Sessa, 2002; Weber et al., 2019) between the hypothesis and a corpus fact. This produces interpretable proofs similar to the compositional entailment trees of prior work (e.g., EntailmentBank (Dalvi et al., 2021)) while tackling the additional challenge of QA.

NELLIE is designed to search across hundreds of trees whose leaves come from one of two types of knowledge: (a) a corpus of semi-structured text statements, such as NL renditions of database entries, or (b) a corpus of free-form NL sentences. Many *correct* proofs might exist for a given hypothesis, but much fewer will be *fully groundable* in the provided corpus, making the search harder than for ungrounded alternatives like Entailer (Tafjord et al., 2022) or COMET-Dynagen (Bosselut et al., 2021). To improve grounding, we introduce two guiding methods to boost the likelihood of generating rules whose conditions match against the available text: (1) For applications where a semi-structured corpus is available, NELLIE leverages this structure via templates to help steer rule generation towards syntax that is more likely to match corpus entries. (2) For both free-form and semi-structured corpora, NELLIE retrieves and conditions on statements to help steer generation towards trees grounded in them. Ablation experiments show these together and individually improve NELLIE's reasoning.

NELLIE expands upon methods that ground known-to-be-true hypotheses to provided facts (SCSEARCH; Bostrom et al. (2022)), (IRGR; Ribeiro et al. (2022)]), extending the paradigm to perform QA. This involves searching for and comparing

trees for conflicting answer options (both true and false). Experiments find NELLIE outperforms a similar-sized state-of-the-art reasoner, Entailer, while producing compositional trees showing how decisions are grounded in corpus facts. We also find NELLIE can exploit both semi-structured and natural text corpora for its reasoning via retrieval and template conditioning. and that the use of templates for rule-generation help reason over semi-structured text data. Our contributions are thus:

1. An architecture for systematic reasoning from a corpus of textual knowledge to answer questions. This can be viewed as a modern approach to expert system inference, but without requiring a formal knowledge representation.

2. An implementation, NELLIE, that outperforms a similar-sized SOTA reasoner (Entailer-3B) while producing grounded trees. To our knowledge, this is the first system to perform grounded XQA as NL entailment tree search.

3.2 Preliminaries

A logical expert system proves a propositional query against a *theory* comprised of facts and inference rules, generally given in the form of Horn clauses. Upon finding a rule whose head can *unify* with the query, a depth-first backward chaining algorithm such as those used in Prolog solvers will perform variable substitution and then recursively attempt to prove the terms in the rule's *body*. For example, a disease classification system might prove query CONTAGIOUS(*flu*)

via facts CONTAGIOUS(*influenza*) and OTHERNAME(*flu*, *influenza*), and conjunctive rule CONTAGIOUS(X) $\Leftarrow$ OTHERNAME(X, Y)$\wedge$ CONTAGIOUS(Y). It does so by unifying CONTAGIOUS(*flu*) with CONTAGIOUS(X) and then recursively unifying the terms in the rule body with their matching facts. Here, *flu* is an object symbol, CONTAGIOUS is a predicate symbol, and X is a variable. See Russell and Norvig (2010) for further details.

3.2.1 Neural Predicates

While most declarative predicates have meaning only in the context of user-defined inference axioms, others can call external neural modules that produce values for their arguments or determine the truth value of the predicate. A popular implementation of this paradigm is DeepProbLog (Manhaeve et al., 2018), which we use to define LM-invoking predicates. In the above example, we might train a sequence-to-sequence (seq2seq) model to produce other names for a disease Y, turning OTHERNAME(Y^+, X^-)[1] into a neural predicate. This mechanism creates the ability to introduce externally defined object symbols, e.g. seq2seq-generated NL, into the engine's vocabulary.

3.2.2 Weak Unification

In classical backward chaining, a *unification* operator assigns equivalence to two logical atoms; this requires atoms to have the same arity and have no unequal ground literals in the same argument position. Issues arise when literals are NL sentences,

[1] In Prolog syntax, '+' denotes inputs, '–' outputs.

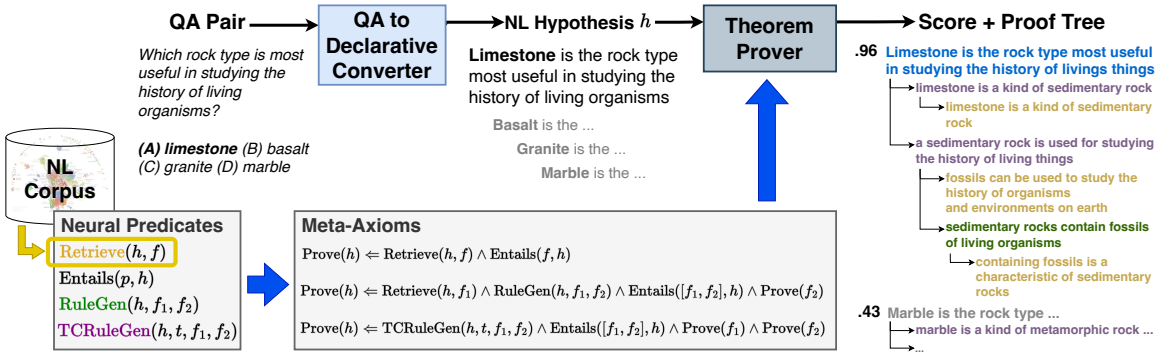


Figure 3.3: NELLIE system framework. An off-the-shelf theorem prover searches for proofs of query PROVE(h), where symbol h is an NL hypothesis translated from a QA pair. The prover uses a set of meta-axioms invoking neural retrieval, entailment, and generation predicates to dynamically instantiate inference rules that use the NL factbase.

which can be syntactically distinct but semantically equivalent. To handle this, Weber et al. (2019) propose a *weak unification* operator, which allows for the unification of *any* same-arity atoms regardless of conflicting symbols.[2] They estimate a *unification score* as the aggregation of pairwise similarity scores using a similarity function $\theta(s_1, s_2) \in [0, 1]$. The score of the full proof is taken as the minimum of scores across all steps. In this work, we apply a similar aggregation; we say that a query fact s_1 "weakly unifies" with provenance fact s_2 with a unification score equal to the confidence of an NLI model taking s_2 as the premise and s_1 as the hypothesis.

[2]Introducing weak unification greatly increases the search runtime, as one might try to unify any two symbols in the vocabulary at every recursive step. This poses a substantial challenge when applying the query grounding algorithm to a search space over NL.

3.3 Overview of Approach

Depicted in Figure 3.3, our framework is comprised of: an external corpus of facts (some $\{f_1, \ldots f_n\}$); a module that converts a QA pair to a hypothesis; an off-the-shelf theorem prover; and a suite of meta-axioms that use neural fact retrieval and dynamic rule generation modules to propose, verify, and score inferences. In my experiments, I consider one implementation of this framework that uses the corpus WorldTree (Xie et al., 2020), a set of 9K NL science facts that can interchangeably be considered as rows in 81 n-ary relational tables.

3.3.1 Question Conversion

Given a multiple-choice question, the system converts each candidate answer into a hypothesis h using a Question to Declarative Sentence model (QA2D; Demszky et al. (2018)) The following is an example conversion by our QA2D model of a multiple-choice question into a set of candidate hypotheses to prove.

Q: Ethanol is an alternative fuel made from corn. What is one of the unfavorable effects of using ethanol as a fuel?

(A) decreased cost of fuel production

(B) decrease in farm land available for food production

(C) increase in the consumption of fossil fuels

(D) increased carbon footprint from driving automobiles

Converted to the following hypotheses:

(A) Using ethanol has an unfavorable effect on the cost of fuel.

(B) Using ethanol as fuel decreases farm land available for food production.

(C) Using ethanol as fuel increases in the consumption of fossil fuels.

(D) The use of ethanol as an alternative fuel can lead to increased carbon footprints.

NELLIE then searches for a proof of each alternative h against its knowledge base. For each alternative, I enumerate p proofs using a time-capped backward chaining search and then take as the system's answer the candidate with the overall highest-scoring proof. This approach is similar to that of Tafjord et al. (2022).

3.3.2 Inference Rule Structure

My approach uses LMs to dynamically generate inference rules given a hypothesis. The rule structure is strikingly simple, instantiating one of the following meta-level templates:

I. `Hypothesis` $\Leftarrow$ `Fact`

II. `Hypothesis` $\Leftarrow$ `Fact1` $\wedge$ `Fact2`

Via template I, the system proves the hypothesis by finding a provenance `Fact` stored in its knowledge store that entails the hypothesis. Via template II, it enumerates a pair, `Fact1` and `Fact2`, both either stored in the knowledge store or themselves recursively proved, such that the pair in conjunction entails the hypothesis.[3] Template I is given higher search precedence than II,[4] yielding an intuitive high-level procedure: we first look up the hypothesis against the factbase, searching for an entailing fact. If we do not find one, we decompose the hypothesis into a pair of statements to be proved. Concretely, for an input hypothesis h, we define the predicate $\textsc{Prove}(h)$ that serves as the primary goal term. We define the following core meta-rules, which use the neural predicates Retrieve, Entails, and RuleGen. At each step in the backward-chaining search, Nellie's Prolog engine attempts to unify a query with the head of one of these three rules:

1. *Fact Unification*

   $$\textsc{Prove}(h) \Leftarrow \textsc{Retrieve}(h^+, f^-) \wedge \textsc{Entails}(f, h)$$

2. *Two Premise Rule Generation*

   $$\textsc{Prove}(h) \Leftarrow \textsc{RuleGen}(h^+, f_1^-, f_2^-) \ \wedge \textsc{Entails}([f_1, f_2], h)$$
   $$\wedge\ \textsc{Prove}(f_1) \wedge \textsc{Prove}(f_2)$$

3. *Retrieved First Premise Rule Generation*

[3] I find that two-premise decomposition is sufficiently powerful and expressive for our purposes. For example, to prove a hypothesis such as those in EntailmentBank (Dalvi et al., 2021) that requires a *three*-premise conjunction $h \Leftarrow f_1 \wedge f_2 \wedge f_3$, Nellie produces instead a recursive set of decompositions $H \Leftarrow f_1 \wedge f_i; f_i \Leftarrow f_2 \wedge f_3$, with the additional interpretability provided by the intermediate NL node f_i.

[4] I only decompose a hypothesis if it's not proved by I.

$$\textsc{Prove}(h) \Leftarrow \textsc{Retrieve}(h^+, f_1^-) \ \wedge \ \textsc{RuleGen}(h^+, f_1^+, f_2^-)$$

$$\wedge \textsc{Entails}([f_1, f_2], h) \ \wedge \ \textsc{Prove}(f_2)$$

3.3.3 Unification with Retrieved Facts

For factbase fact f, $\textsc{Prove}(f)$ is vacuously true. Rule 1 shows how we prove $\textsc{Prove}(h)$ using retrieval. The predicate $\textsc{Retrieve}$ proposes candidate f_i's given h using a FAISS (Johnson et al., 2019)-based nearest neighbor dense retrieval index. I train the retrieval encoder via ranking loss such that the embedding for a hypothesis is maximally similar to its supporting facts. To promote logical coherence and improve the precision of the system, I apply a set of neural models for recognizing textual entailment (RTE) as filters $\textsc{Entails}_j(\cdot)$ that iteratively rule out f_i candidates that are not classified as entailing h.

$$\textsc{Entails}(f_i, h) \Leftarrow \bigwedge_{j=1...n} \textsc{Entails}_j(f_i, h)$$

Implicit in rule 1 is that $\textsc{Prove}(h)$ weakly unifies with some $\textsc{Prove}(f_i)$; I assign the unification score $\theta(h, f_i)$ equal to the confidence of one RTE model.

For some questions such as the one depicted in Figure 3.4, it is necessary to ground a subquery in evidence from the problem. To handle this, I add the question setup (defined as all but its last sentence) as a "fact" always proposed by $\textsc{Retrieve}$.

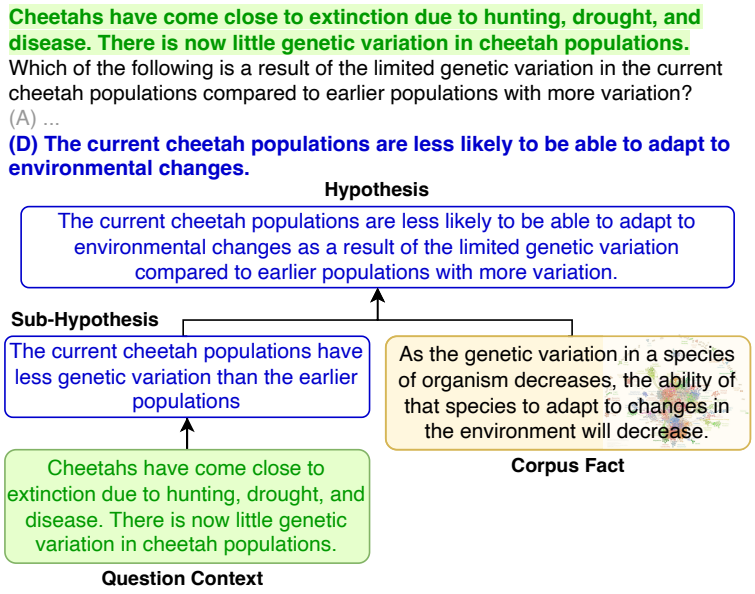


Figure 3.4: Example question in which a sub-hypothesis in the proof tree is grounded to the question context rather than to the factbase.

3.3.4 Dynamic Rule Generator (DRG)

If we do not find an entailing fact for hypothesis h, we decompose h into a pair of entailing premises; we propose candidates using nucleus sampling (Holtzman et al., 2019) from a seq2seq model. The predicate RuleGen(h, f_1, f_2) prompts a model trained to generate $h \rightarrow f_1, f_2$ pairs. The space of potential decompositions is very large; there are many deductively valid ways to prove a hypothesis in natural language, though only a fraction of them are ultimately groundable in the provided factbase. To bias the proof search towards those more likely to be grounded in the corpus, I adopt a two-pronged approach, illustrated in inference rules 2 and 3, that proposes a *heterogeneous search frontier* using different biasing strategies in addition to straightforward LM sampling.

3.3.5 Template Conditioned Generation (TCG)

One way in which we improve our LM-based proposal function is to leverage the high-level structure that supports reasoning in a given domain: we propose *template-guided generation* to bias search towards the semi-structure of WorldTree.[5] WorldTree tables correspond to types of facts with similar syntax and semantics that support scientific reasoning. WorldTree questions are annotated with the fact rows that support their answers. Each table can be viewed as an n-ary relation of columns whose values are text spans. E.g., the Taxonomic relation has columns *<A>*, *[HYPONYM]*, *<is a / a kind of>*, *<SCOPE1>*, *[HYPERNYM]*, *<for>*, *[PURPOSE]*. Rows include *'a bird is an animal'* and *'a seed is a kind of food for birds.'*

Thus, for the RuleGen predicate in rule 2, half of the f_1, f_2 candidates are sampled from the DRG conditioned on h plus a template that cues the model to reflect the syntax of a WorldTree table.[6] We train the DRG to accept any masked infilling template (e.g. *<mask>is a kind of <mask>*, akin to those used to pretrain LMs (Lewis et al., 2019; Raffel et al., 2020)), and propose decompositions whose first fact reflects the template's syntax. We thus create a $h, t_1 \rightarrow f_1, f_2$ model. We feed the model templates drawn from WorldTree's tables, guiding it towards proof steps

[5]While our approach is applicable to a wide array of reasoning problems, the use of WorldTree-specific templates illustrates one way by which we can infuse domain-specific structure into the neural search algorithm. This is a strict departure from the burdensome process of symbolic knowledge engineering. As our experiments show, this guidance method improves Nellie's QA accuracy by a few points, though ablating it yields a similarly strong performance.

[6]This intuition follows from the observation that over 85% of the non-leaf nodes in the trees of EntailmentBank (Dalvi et al., 2021), a dataset of gold trees rooted in WorldTree facts, also reflect the syntax of a WorldTree table.

more likely to be grounded in the factbase. Figure 3.5 depicts the infilling templates gathered from WorldTree and used for template-guided rule generation. We manually annotated this list. The full set of templates is of size 150. We reuse the same model for non-template-conditioned generation by feeding it an empty template.[7] In practice, we make two generation calls: one samples m free-generated candidates, and a second samples n candidates for each of n_t templates.

$$\textsc{RuleGen}(h^+, f_1^-, f_2^-) \Leftarrow \textsc{Member}(t, \textsc{Templates} \cup \{\texttt{BLANK}\}) \\ \wedge \textsc{TCRuleGen}(h^+, t^+, f_1^-, f_2^-)$$

3.3.5.1 Template Selection

WorldTree is a diverse corpus, containing tables that are specific to a particular subset of science problems (e.g. the "Predator-Prey" table). Generating decompositions according to these templates in a context-independent manner risks wasting time and compute to consider inference patterns that are almost certainly irrelevant and unviable. As conditioning on dozens of templates is also computationally expensive, we introduce a *case-based reasoning* (Schank, 1983; Das et al., 2021) approach that selects relevant templates for a given hypothesis. We construct an Okapi-BM25 (Jones et al., 2000) retrieval index over questions from the WorldTree QA training set to obtain the most lexically similar items to a query hypothesis. At inference time, we

[7] Though a separate model could be used for vanilla generation in the same inference pipeline.

WorldTree Relation	**Template**
INSTANCES	<mask>is <mask>
ACTION	<mask>to <mask>
KINDOF	<mask>is a kind of <mask>
PROP-THINGS	<mask>are <mask>
ACTION	<mask>for <mask>
AFFORDANCES	<mask>can <mask>
IFTHEN	if <mask>then <mask>
PROP-THINGS	<mask>has <mask>
CAUSE	<mask>causes <mask>
MADEOF	<mask>made of <mask>
PROP-THINGS	<mask>have <mask>
REQUIRES	<mask>requires <mask>
ACTION	<mask>is when <mask>
USEDFOR	<mask>used to <mask>
PARTOF	<mask>a part of <mask>
CHANGE	<mask>changes <mask>
USEDFOR	<mask>used for <mask>
ATTRIBUTE-VALUE-RANGE	<mask>means <mask>
AFFECT	<mask>has <mask>impact on <mask>
PROP-ANIMAL-ATTRIB	<mask>is a <mask>animal
SOURCEOF	<mask>is a source of <mask>
CONTAINS	<mask>contains <mask>
COUPLEDRELATIONSHIP	as <mask>will <mask>
CHANGE-VEC	<mask>decreases <mask>
COMPARISON	<mask>than <mask>
ACTION	<mask>is when <mask>to <mask>
CAUSE	<mask>can cause <mask>
LOCATIONS	<mask>found <mask>
COMPARISON	<mask>more <mask>
PROP-INHERITEDLEARNED	<mask>is <mask>characteristic
DURING	<mask>during <mask>
CHANGE-VEC	<mask>increases <mask>
AFFORDANCES	<mask>can <mask>by <mask>
EXAMPLES	an example of <mask>is <mask>
SOURCEOF	<mask>produces <mask>
CONTAINS	<mask>contain <mask>
CHANGE	<mask>converts <mask>
COMPARISON	<mask>is <mask>than <mask>
FORMEDBY	<mask>formed by <mask>
CONSUMERS-EATING	<mask>eat <mask>
SOURCEOF	<mask>provides <mask>
PROP-RESOURCES-RENEWABLE	<mask>is <mask>resource
ATTRIBUTE-VALUE-RANGE	<mask>means <mask>of <mask>

Figure 3.5: Partial list of WorldTree templates used for guided generation, sorted by frequency of occurrence in EntailmentBank trees (full list omitted for space).

select the top-k most similar questions to the query and take as our template subset the tables of the questions' annotated facts.

3.3.6 Retrieval Conditioned Generation (RCG)

In rule 3, rather than generate a pair of subqueries, we *immediately ground* half of the antecedent by choosing as f_1 a fact retrieved directly from the corpus. We have the DRG force decode f_1 before generating f_2 as normal, then recur only on f_2. In practice, we retrieve some top-n_f candidate f_1's scored by a retrieval bi-encoder for each fact and generate candidate decomposition sets in parallel for each fact. The number of decompositions per fact is determined by the number of generations that can be made in parallel on a given GPU, as we enforce that all retrieval-conditioned generations happen simultaneously (one call to the Hugging Face `generate` call, which can take 7-10 seconds in some cases). Given a GPU-specific cap on parallel generations C, we generate C/n_f decompositions per support fact. In our case, using a 24GB RTX GPU, the cap C is set to 40 for a T5-3B-based decomposition generator.

3.3.7 Filters

As stochastic sampling from LMs can be noisy, some fraction of the generated candidate set may be invalid: premises may be incoherent, or the decomposition might not properly entail h. Accordingly, we introduce a set of compositional entailment

verifiers (Khot et al., 2020; Jhamtani and Clark, 2020) trained on two-premise compositional entailment. We also add a "self-ask" filter, which following Tafjord et al. (2022) is an LM fine-tuned to assign a statement a truth value. If the confidence is below 0.5 for either an entailment judgment or the 'self-ask' belief in f_1 or f_2, then we filter the pair. All filters condition on the question text as context. When NELLIE uses these rules, the unification score equals the lowest of the scores for PROVE(f_1), for PROVE(f_2), and the confidence of entailment filters $s_e([f_1, f_2] \Rightarrow H)$.

3.3.8 Proof Search

Given a query, NELLIE searches for t seconds to find up to p proofs of depth d or less. We follow Weber et al. (2019) in pruning search branches whose unification score is guaranteed to fall below the current best, given our monotonic aggregation function min($\cdot$). Found proofs that score under the current best do not count towards p. Figure 3.6 illustration of a successful search trace combining the modules described above. Full pseudocode for the algorithm, which follows a depth-first search with a breadth-first lookahead (Stern et al., 2010) to check for the unification of generated subgoals, can be found in Algorithm 1. The logic of the algorithm is implemented in a meta-interpreter coded in DeepProbLog (Manhaeve et al., 2018). It is parameterized by

1. A maximum number of proofs m at which to cut off searching. In experiments, we set this to 10 for top-level queries and 2 for recursive subqueries.

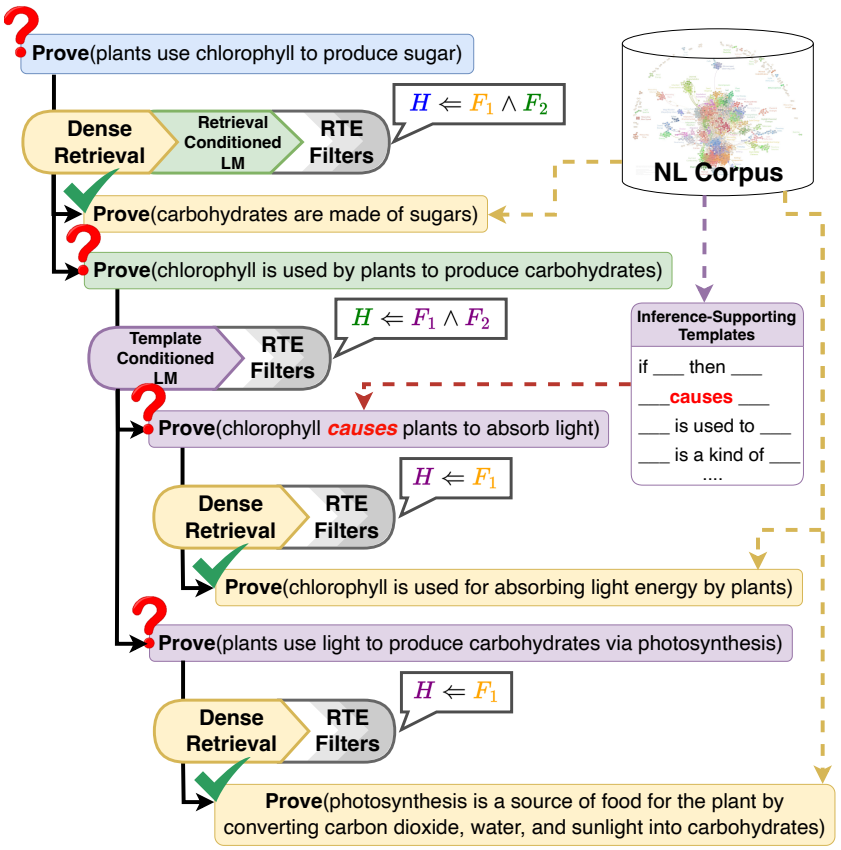


Figure 3.6: Given a query, NELLIE performs a neuro-symbolic backward chaining search for proof trees whose leaves are grounded in a corpus of facts. It generates candidate decomposition rules guided by retrieved facts or templates. Then, it recursively tries to prove rule conditions via entailment from the corpus or further decomposition.

2. A number of support facts n_f to retrieve at each call to RETRIEVE_K, which we set to 15.

3. Candidate generation rates n_v for vanilla nucleus-sampled decompositions, n_t for template-conditioned decompositions, and n_r for retrieval-conditioned generations. We set these each to 40.[8] Upon removing exact match duplicates,

[8]Due to batching, these correspond to 3 total calls to the HuggingFace library's `generate` function for the T5-3B model.

a call to RuleGen produces about 100 candidates.

4. Entailment scoring module $\textsc{Score}_e(\cdot)$, which is a separate RTE cross-encoder model for single- and double-premise entailments.

Algorithm 1 NELLIE's backchaining algorithm, coded in Prolog, for searching for proof list $tree^{(1)}(H), \ldots tree^{(m')}(H)$ with scores $s^{(1)}(H), \ldots s^{(m')}(H); m' \leq m$ for a hypothesis H against factbase K. The rule generator produces n_v candidates via vanilla nucleus sampling, n_t candidates via sampling from templates, and n_r candidates whose first fact is retrieved from K. The search is capped at max depth $d_{\max}$. All GENERATE, RETRIEVE, and entailment scorer SCORE_e calls in **forall** loops are batched. All MODULE outputs are cached. ONEPREMISEFILTERS and TWOPREMISEFILTERS are ordered lists of neural entailment models. $\vdash$ denotes weak entailment.

procedure PROVE(H, m, d) → proofs $tree^{(1)}(H), \ldots tree^{(m)}(H)$ & scores $s^{(1)}(H), \ldots s^{(m')}(H)$:
  i. **check weak unification** against K:
    entailing premises P = PREMISELOOKUP(H)
    if P nonempty
    then return proofs= [$(p \vdash H)$ for $p \in P$], scores= [$\text{SCORE}_e(p \vdash H)$ for $p \in P$]
    elif $d = d_{\max}$
    then Fail
  ii. **generate candidate decompositions:** *// If H does not unify against K, decompose it*
    decompositions D = RULEGEN(H)
  iii. **check whether any decompositions pairs both unify:** *// this step amortizes premise lookups*
    proofs = []; scores = []
    forall $(p_1, p_2) \in D$ **do**
      if $p_1 \in K$ **and** $\exists p_2' \in$ PREMISELOOKUP(p_2) *// if p_1 already grounded, its subtree is leaf*
      then proofs += $(\{p_1, (p_2' \vdash p_2)\} \vdash H)$
        scores += min($\text{SCORE}_e(p_2' \vdash p_2), \text{SCORE}_e(p_1 \wedge p_2 \vdash H)$) *// tree score is min of scores used to construct it*
      elif $\exists p_1' \in$ PREMISELOOKUP(p_1) **and** $\exists p_2' \in$ PREMISELOOKUP(p_2)
      then proofs += $(\{(p_1' \vdash p_1), (p_2' \vdash p_2)\} \vdash H)$
        scores += min($\text{SCORE}_e(p_1' \vdash p_1), \text{SCORE}_e(p_2' \vdash p_2), \text{SCORE}_e(p_1 \wedge p_2 \vdash H)$)
    if proofs not empty
    then return proofs, scores
  iv. **recur on generated premises:** *// if still no proof, need to recur*
    proofs = []; scores = []
    forall $(p_1, p_2) \in D$ **do**
      if $\text{SCORE}_e(p_1 \wedge p_2 \vdash H) >$ max(scores) **and** len(proofs) $< m$ *// only recur if chance of better score*
      then $(\text{proof}_{p_1}, S_{p_1})$ = PROVE($p_1, 1, d+1$)
        $(\text{proof}_{p_2}, S_{p_2})$ = PROVE($p_2, 1, d+1$)
        recursive score $S_r = \min(S_{p_1}, S_{p_2}, \text{SCORE}_e(p_1 \wedge p_2 \vdash H))$
        if $S_r >$ max(scores)
        then proofs += $(\{\text{proof}_{p_1}, \text{proof}_{p_2}\} \vdash H)$
          scores += S_r
    return proofs, scores

procedure PREMISELOOKUP(H) → premises P s.t. $\forall p \in P, p \vdash H$ *// try to ground H in K via weak entailment*
    candidate premises $P = \text{RETRIEVE}_K(H, n_f)$ *// retrieve candidate premises s.t. $p \vdash H$*
    forall $f \in$ ONEPREMISEFILTERS **do**
      $P = f(P, H)$ *// only keep premises that pass all filters*
    return P

procedure RULEGEN(H) → decompositions $D = (p_1^1, p_2^1), \ldots (p_1^c, p_2^c)$ *// find candidate pairs s.t. $p_1 \wedge p_2 \vdash H$*
    D = []
    D += GENERATE($H, n = n_v$)
    WTTEMPLATES = CASEBASEDRETRIEVETEMPLATE(H)
    outputs per template $o_t = n_t/\text{len}(\text{WTTEMPLATES})$
    forall $t \in$ WTTEMPLATES **do** *// generate candidates for each of 50 templates in WTTEMPLATES*
      D += GENERATE($[H\ t], n = o_t$) *// T5 model accepts optional t appended to H*
    retrieved first premises $P_1 = \text{RETRIEVE}_K(H, n_f)$
    outputs per premise $o_r = n_r/\text{len}(P_1)$
    forall $p_K \in P_1$ **do** *// generate candidates with retrieved support facts as first premise*
      D += GENERATE(H, prefix = $p_K, n = o_r$) *// Force decode p_K then generate a p_2*
    forall $f \in$ TWOPREMISEFILTERS **do** *// includes entailment, regex, self-asking filters*
      $D = f(D, H)$ *// only keep premise pairs that pass all filters*
    return D

3.3.9 Model Details and Training

We train the components of NELLIE to be able to answer questions in the Science QA domain. The different neural modules are trained on reformulations of existing datasets for scientific reasoning. We use the following sources: **WorldTree Explanations (Xie et al., 2020)** and **EntailmentBank (Dalvi et al., 2021)** are used as described in the paper. **eQASC (Jhamtani and Clark, 2020)** and **SciTaiL (Khot et al., 2018)** are datasets of two- and one-premise entailment questions.

3.3.9.1 QA2D

Our QA2D model is a T5-11B (Raffel et al., 2020) fine-tuned by Tafjord et al. (2022) on the original QA2D dataset (Demszky et al., 2018).

3.3.9.2 Retrieval Module

The dense retrieval model is a SentenceTransformers Siamese BERT-Network encoder (Reimers and Gurevych, 2019) pretrained on the MS MARCO IR corpus (Nguyen et al., 2016), which we fine-tune via ranking loss to maximize the cosine similarity between a hypothesis and its support facts. I gather h, f_1 from the WorldTree, EntailmentBank, and eQASC training sets, then sample random negatives during training. While it is difficult to evaluate the retrieval module's performance in the process of NELLIE's multi-hop question answering (during which the system

makes dozens of retrieval calls returning different retrieval sets), I benchmarked the fine-tuned cross-encoder on its own using a series of gold-annotated support fact sets from WorldTree. For each question and correct hypothesis in the EntailmentBank test set, I evaluated against the following gold sets:

1. The facts leaves appearing in the gold-annotated entailment tree

2. The facts from WorldTree expert-annotated by Jansen et al. (2021) to have an ordinal relevance score of 5 or more on their scale from 0-6. 5/6 generally denotes "core" facts.

3. The facts annotated by Jansen et al. (2021) to have a score of 3 or more. This generally denotes at least "important" facts.

	EB Gold Leaves		Core facts		Important facts	
	Dev	Test	Dev	Test	Dev	Test
R@1	0.19	0.15	0.24	0.20	0.09	0.07
R@3	0.41	0.30	0.47	0.39	0.21	0.16
R@5	0.52	0.41	0.58	0.49	0.29	0.23
R@10	0.67	0.51	0.72	0.61	0.43	0.33
R@15	**0.73**	**0.58**	**0.79**	**0.66**	**0.50**	**0.40**
R@20	0.78	0.63	0.83	0.71	0.56	0.45
R@25	0.80	0.66	0.86	0.75	0.60	0.50
R@30	0.82	0.70	0.87	0.79	0.64	0.54
R@50	0.86	0.77	0.92	0.85	0.73	0.63

Table 3.1: Recall of support fact sets with question text plus correct answer hypothesis as query. The R@15 row is bolded because I chose $n_f = 15$ via Optuna-based hyperparameter search.[9]

3.3.9.3 Dynamic Rule Generator

The DRG is a T5-3B model (Raffel et al., 2020) fine-tuned on $h \rightarrow [f_1, f_2]$ pairs drawn from eQASC, which are numerous (26K) but noisy, and the binary composition steps from the EntailmentBank training set, which are few (3.8K) but ultimately rooted in facts from WorldTree. To make the DRG able to condition on templates, I adopt the span masking strategy from Raffel et al. (2020) to create a randomly masked version $\hat{f}_1$ of each f_1 and then train the model on $h, \hat{f}_1 \rightarrow f_1, f_2$ pairs. For each original triplet, I create one training example with a template, and one in which $\hat{f}_1$ is an empty <mask>, giving the model the dual capacity for non-template-generation.

3.3.9.4 Entailment Filters

I found that a mixture of strategies helps prioritize precision over recall when filtering candidates. For 1-premise filtering for unification and 2-premise filtering for rule generation, I fine-tune a separate pair of models consisting of a SentenceTransformer cross-encoder classifier and a T5 seq2seq. The former is trained via classification loss, the latter via sequence cross entropy loss. The 1-premise filters are trained on the SciTail dataset, the 2-premise filters on eQASC and EntailmentBank. For 2-premise entailment, I additionally add a "self-ask" filter, which following Tafjord et al. (2022) is a T5 model fine-tuned to produce its 0-to-1 confidence that a statement is true. If the confidence is below 0.5 for either f_1 or f_2, I filter the pair. As in Tafjord et al. (2022), I train our DRG seq2seq in a multitask fashion so that it serves as both the

rule generator as well as the "self-ask" filter and (half of) the 2-premise entailment filter.

3.3.9.5 Training

I use the default training parameters from the SentenceTransformers library[10] in order to train CrossEncoder-based classifier and Bi-encoder retriever models for 3 epochs. I correspondingly use the default fine-tuning parameters from the HuggingFace library in order to train our T5-based generation and classification models.

3.4 Experiments

Our experiments illustrate how the approach exemplified by NELLIE provides grounded and logical explanations, performing comparably or better than approaches that do not satisfy these properties. Our task metric is accuracy: whether a generated proof of the correct option outscores any other.[11]

3.4.1 Evaluation Datasets

We evaluate models on two multiple-choice QA datasets constructed so that correct answers are supported by facts in the WorldTree corpus:

[10] https://tinyurl.com/mvhbmfcc
[11] If it produces no proofs, it gives no answer and is wrong.

3.4.1.1 EntailmentBank

EntailmentBank (Dalvi et al., 2021) is a dataset of entailment trees for declarativized answers to the AI2 Reasoning Challenge (ARC) dataset (Clark et al., 2018), showing how the hypothesis can be derived via compositional entailment hops from WorldTree facts. We recast the test set, initially designed to test tree reconstruction, into QA by retrieving the corresponding multiple-choice ARC questions from which the hypotheses were constructed.

3.4.1.2 WorldTree

WorldTree (Xie et al., 2020) is a subset of the ARC dataset annotated with undirected explanation graphs whose nodes are facts from the WorldTree tablestore. We note that WorldTree explanations do not show how facts should combine. *There is no guarantee that fully grounded trees exist for these questions using the WorldTree corpus alone.* The difficulty of this task is akin to that of a course exam: the teacher (us) provides the student (the model) with a very large study guide of facts, but expects the student to **use** these facts by composing them to reason coherently about a problem.

3.4.2 Hyperparameters

NELLIE searches for up to p=10 proofs of max depth d=5 with a timeout of t=180 seconds per option. The DRG produces 40 (pre-filter) candidates divided evenly

amongst the set of templates drawn via case-based retrieval from the top-(k=10) most similar questions to a given query hypothesis. It also generates 40 free-generated candidates, and 40 retrieval-conditioned ones (4 each for the 10 top-scoring retrieved f_1's). It therefore considers 120 candidate decompositions at each recursion step. All sampling is done via nucleus sampling ($p = .95$). RTE models filter all candidates for which the classification softmax likelihood of entailment is less than 0.7. We remove duplicates before filtering.

I used Optuna (Akiba et al., 2019) to find the number of facts for retrieval-conditioned generation, the p value for nucleus sampling, the entailment threshold, and the number of candidates generated at each recursion step. I used as the objective the overall QA accuracy on the EntailmentBank dev set. I ran it for about 300 trials, and found that the most important parameters were the entailment threshold (too low = too many bad proofs, too high = no proofs) and the nucleus p (less than 0.8 reduced QA performance by 10-15. Experiments were run on 24GB Quadro RTX 6000s.

3.4.3 Baselines

I evaluate NELLIE against **Entailer** (Tafjord et al., 2022), another system that produces entailment tree proofs via backward chaining. Entailer stops recurring **not** when it finds entailing facts from a corpus, but rather when *the model believes* that a subquery is true with high confidence. Its trees are thus not grounded in verified facts. We reimplemented their algorithm using their T5-11B-based model, reporting their

configuration of a max tree depth (**D**) of 3 and minimum of 1 (i.e. ≥ 1 decomposition). As their work found that the maximum depth is *inversely proportional* with QA performance, we also evaluate their ablated setting which recurs exactly once (D=1). This is a baseline that is neither grounded nor particularly interpretable, generating just a pair of statements. To isolate the impact of model size, we also evaluate an **Entailer-3B** model based on the same T5-3B model as NELLIE's DRG.[12][13]

I also compare against grounded xQA methods without logical structure (see Figure 3.2): **PathNet** (Kundu et al., 2019), which constructs 2-hop chains by linking entities between facts, and three approaches that build graphs from facts using linear programming: **TupleILP** (Khot et al., 2017), **ExplanationLP** (Thayaparan et al., 2021), and ***Diff*-Explainer** (Thayaparan et al., 2022).[14]

3.4.4 Results

Table 3.2 shows QA performance. NELLIE's overall (**Ovr**) accuracy of over 70% matches that of the best-performing grounded baseline, *Diff*-Explainer, on WorldTree, and is within 2 points of the best logically directed baseline, Entailer-11B. This is notable given Entailer-11B is a much larger model and has no requirement to provide grounded proofs. NELLIE outperforms the same-sized Entailer-3B baseline by a margin

[12]I obtained the training data from Tafjord et al. (2022) and trained our model in consultation with the authors.

[13]I also experimented with an 11B model as NELLIE's DRG; however this model performed very poorly, timing out very frequently given the increased duration of generate calls on available GPUs.

[14]We show results from Thayaparan et al. (2021) & Thayaparan et al. (2022).

	Explanations		QA Accuracy (%)		
	Grounded	**Logical**	**Overall**	**Easy**	**Challenge**
EntailmentBank QA					
NELLIE (3B)	**Yes**	**Yes**	71.4	76.4	60.4
Entailer-3B					
(D = 3)	No	**Yes**	48.7	52.8	39.6
(D = 1)	No	No	64.9	71.2	50.9
Entailer-11B					
(D = 3)	No	**Yes**	73.2	77.3	64.2
(D = 1)	No	No	71.1	76.4	59.4
WorldTree QA					
NELLIE (3B)	**Yes**	**Yes**	71.4	75.7	60.9
Entailer-3B					
(D = 3)	No	**Yes**	45.2	50.1	33.3
(D = 1)	No	No	47.7	51.6	38.5
Entailer-11B					
(D = 3)	No	**Yes**	73.2	76.7	64.6
(D = 1)	No	No	74.1	78.7	63.0
PathNet	**Yes**	No	43.4		
TupleILP	**Yes**	No	49.8		
ExplanationLP	**Yes**	No	62.6		
Diff-Explainer	**Yes**	No	71.5		

Table 3.2: NELLIE performance vs. comparable XQA systems.

of 22% on EntailmentBank and 26% on WorldTree under its default max D = 3 configuration, and even outperforms the less explainable D = 1 variant.

Figure 3.7 shows the results of ablating NELLIE's two conditional generation modules and replacing them with vanilla generation. Removing both template- and retrieval-conditioned generation lowers NELLIE's performance in all circumstances, highlighting the empirical benefit of structured guidance. Ablating TCG ((– TCG),

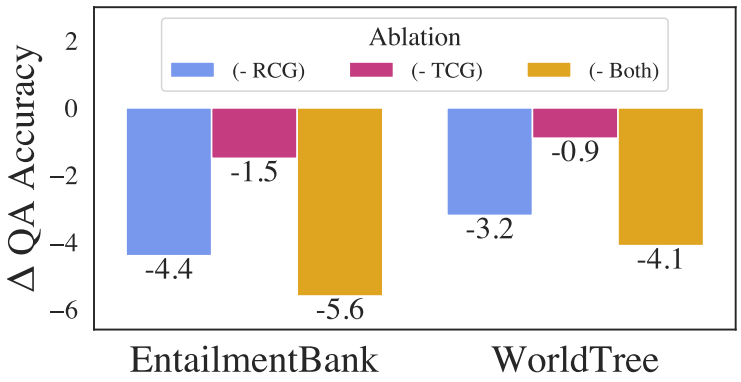


Figure 3.7: Effect of ablating one or both of rule-conditioned (RCG) and template-conditioned generation (TCG).

and the drop from (− RCG) to (− Both)) reduces performance by 1-1.5 points. Ablating RCG (NELLIE vs (− RCG)) drops it by 3-4.

3.4.5 Domain Generalization and Knowledge Scaling

While NELLIE was trained on datasets centered around WorldTree, we show that it can perform in another domain by altering the knowledge over which it reasons. We consider OpenBookQA (Mihaylov et al., 2018), a dataset that requires reasoning over facts not included in WorldTree; each question is associated with one WorldTree science fact ("metals conduct electricity"), but also one fact from a separate pool of common knowledge ("a suit of armor is made of metal").

As with a classical expert system, we have designed NELLIE to be "improvable merely by making statements to it" (McCarthy, 1959). This suggests that as we increase the provided knowledge store, we should expect NELLIE to reason more

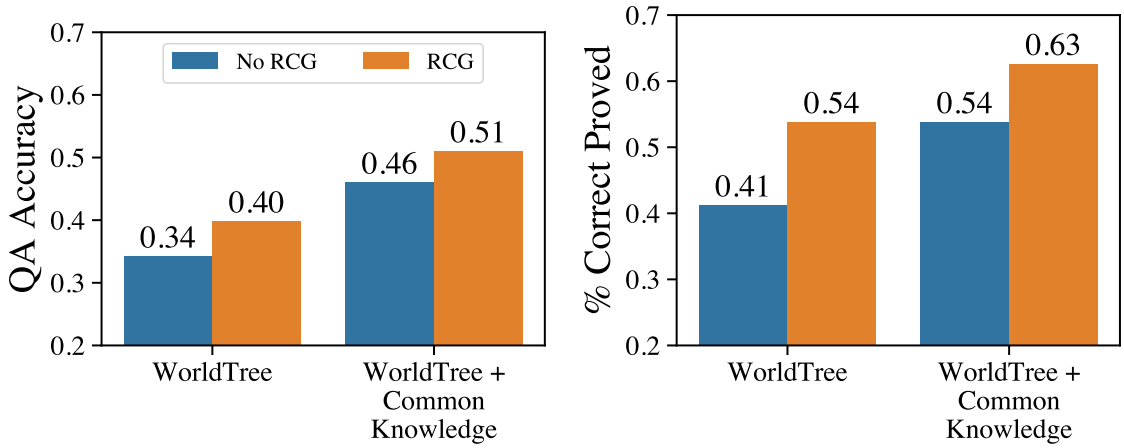


Figure 3.8: NELLIE QA accuracy and proof recall on OBQA with vs. without access to common knowledge statements

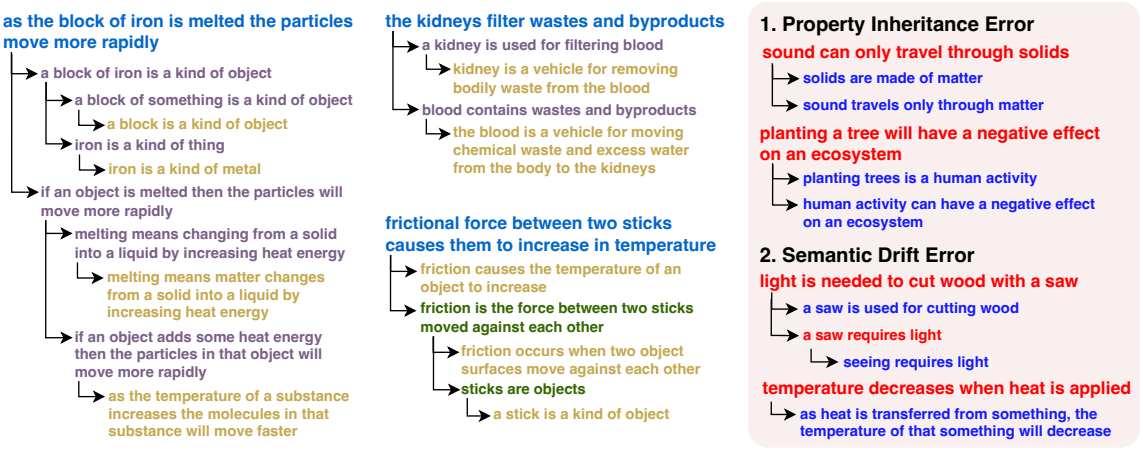


Figure 3.9: **(Left)** Example NELLIE proofs. Top-level queries are decomposed into subqueries via retrieval- or template-conditioned generation. Proof leaves are corpus facts. **(Right)** Common classes of error causing false statements to be grounded in true ones.

effectively. Because WorldTree does not cover all the knowledge required to answer OBQA questions, we test whether NELLIE performs better on them when we add the common knowledge to its retrieval index. As the common knowledge annotations are one fact per question and are not designed specifically for entailment, it is unlikely

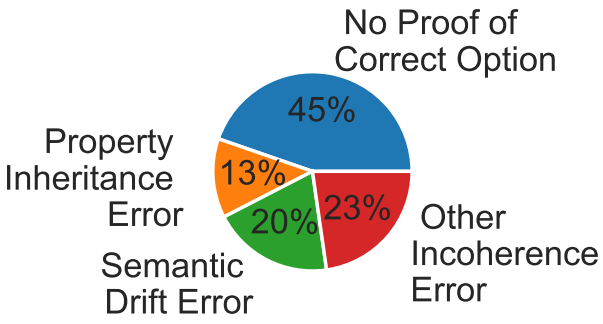


Figure 3.10: Distribution of NELLIE error classes among 136 incorrect answers on EntailmentBank questions

a priori that a fully grounded entailment tree can be created using the available knowledge alone, making this a challenging task.

Figure 3.8 shows NELLIE's accuracy with and without RCG. We do not use TCG, as the targeted common knowledge is free-form. We find that QA accuracy increases 11-12% with the common knowledge added. The percent of correct statement proved also increases 9-12%. These results also highlight the importance of RCG, as it provides a 5% QA boost and 13% on correct proof rate. These results show that NELLIE can be applied out-of-the-box to a dataset requiring reasoning over a different source of free-form NL knowledge.

3.4.6 Tree Error Analysis

We find that NELLIE can produce high quality proofs of correct hypotheses; examples are shown in Figure 3.9 (left) and Figure 3.11.

However, the system can also generate proofs for incorrect answers that bypass its

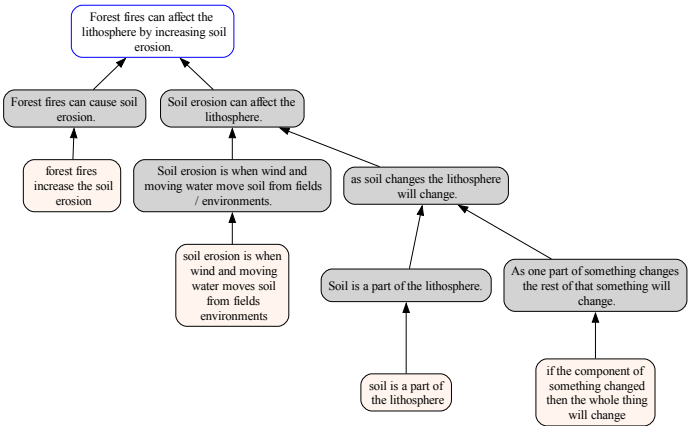
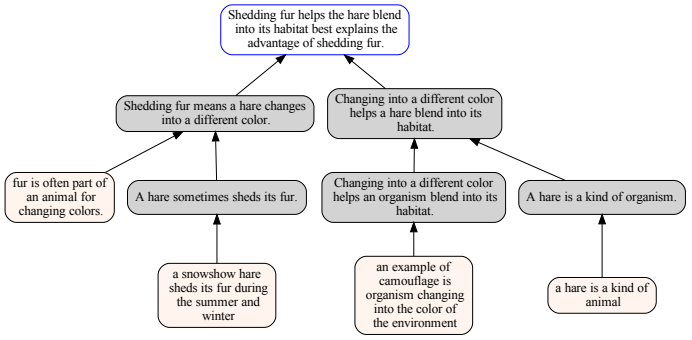
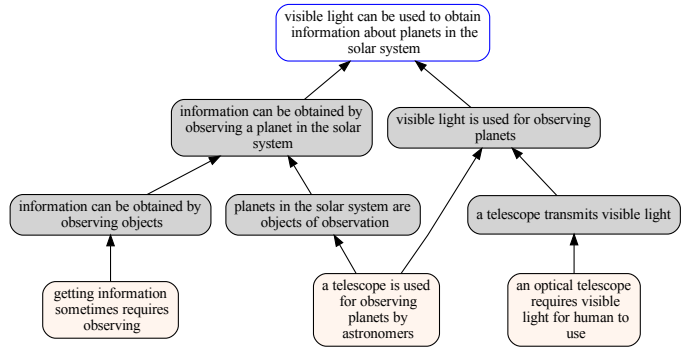
Question	Nellie Proof
In what way can forest fires affect the lithosphere? **(A) increasing soil erosion** (B) causing air pollution (C) increasing seismic activity (D) causing violent dust storms	
The snowshoe hare sheds its fur twice a year. In the summer, the fur of the hare is brown. In the winter, the fur is white. Which of these statements best explains the advantage of shedding fur? (A) Shedding fur keeps the hare clean. (B) Shedding fur helps the hare move quickly. (C) Shedding fur keeps the hare's home warm. **(D) Shedding fur helps the hare blend into its habitat.**	
Scientists use large optical telescopes to obtain information about the planets in the solar system. What wavelengths of electromagnetic radiation provide this information? (A) gamma radiation (B) infrared radiation (C) radio waves **(D) visible light**	

Figure 3.11: Example multiple-choice questions with Nellie's answer and corresponding proof. Blue-outlined nodes are hypotheses generated by the QA2D model for the correct answer highted in **green**. Grey nodes are generated subqueries; tan nodes are leaves from Nellie's knowledge corpus.

entailment filters. We can diagnose (and perhaps annotate) error patterns to address in future work. We list a few categories, depicted in Figure 3.9 (right). A common pattern of error is **Hypernym Property Inheritance Errors** (also known as the "fallacy of the undistributed middle"), in which the model assumes that a member of a taxonomic class has a property of their hypernym, but the property is not universal. A colloquial example is inferring *penguins can fly* from *penguins are birds* & *birds can fly*. We also observe an amount of **Errors from Semantic Drift** (Khashabi et al., 2019) between inference hops, culminating in RTE model false positives. E.g., in block 2 of Figure 3.9 (right), which object loses temperature during heat transfer changes between hops. Figure 3.10 shows the distribution of these errors on a set of 136 incorrect answers from EntailmentBank. The predominant error case is a failure to find any proof of the correct option.

3.4.7 Correct Answer Tree Analysis

To investigate whether NELLIE reaches a right solution with right vs wrong reasoning, we manually inspected for reasoning errors in 50 correct answer trees produced by NELLIE. We found that 39/50 (78%) of trees were perfectly acceptable. Most of the 11 unacceptable trees were due to incompleteness at one decomposition step. In Chapter 4 I more comprehensively how entailment tree-based reasoners can be right for the wrong reasons, and explore mechanisms for measuring and addressing this challenge.

3.5 Impact of Design Choices

NELLIE is comprised of numerous modules that are very difficult to evaluate in isolation given a lack of in-domain test sets for tasks like QA2D, decomposition generation ($H \rightarrow F1, F2$) compositional RTE ($H \Leftarrow F1, F2$), single-premise RTE ($H \Leftarrow F1$), and fact retrieval. Below, I enumerate some modeling choices, alternatives I tried, and the extent to which I observed them impact the system.

- **QA2D:**

  NELLIE uses: Tafjord et al. (2022)'s T5-11B model.

  We also tried: using GPT-J (Wang and Komatsuzaki, 2021a) with in-context-learning exemplars drawn from EntailmentBank. This results in **10+ point drop in QA**, mainly because (A) EntailmentBank hypotheses are designed to be *generics* instead of *question-conditioned statements* and (B) GPT-J tended to generate true statements even for wrong answer options.

- **Decomposition generation ($H \rightarrow F1, F2$):**

  NELLIE uses: A T5-3B model trained in a multi-angle fashion similar to Tafjord et al. (2022), using additional data for compositional entailment and decomposition generation.

  We also tried: Using a T5-Large model trained only on the decomposition data. This allowed us to generate far more decomposition candidates per hypothesis, but rate of obtaining a *good* decomposition was far lower. This led to *5+ point*

drop in QA.

- **Compositional RTE ($H \Leftarrow F1, F2$):**

  NELLIE uses: A sequence of (1) The same T5-3B model as for generation and (2) a cross-encoder trained on eQASC and EntailmentBank.

  We also tried: (A) using a bi-encoder trained via contrastive loss (this was not as effective at ruling out bad decompositions), (B) not training the bi-encoder to condition on the question, and (C) only training on EntailmentBank (as Tafjord et al. (2022) do). All three variants reduce QA by 3-7%.

- **Single-premise RTE ($H \Leftarrow F1$):**

  NELLIE uses: (1) A cross-encoder trained on Sci-Tail

  We also tried: an off-the-shelf RTE model,[15] which had too high a false positive rate to be effective.

- **Fact Retrieval:**

  NELLIE uses: a Sentence-Transformers bi-encoder pretrained on MS MARCO and fine-tuned on $(H, F1)$ pairs drawn from WorldTree, EntailmentBank, and eQASC.

  We also tried: the same bi-encoder off-the-shelf without finetuning. This only dropped performance by 3-4 points.

[15] https://huggingface.co/cross-encoder/nli-deberta-v3-base

3.5.1 Development Timeline

The performance shown in Table 3.2 was obtained over the course of a 9-month period of experimentation and development. Below I review key revisions to the design of the system that led to its end performance shown in Figure 3.12.

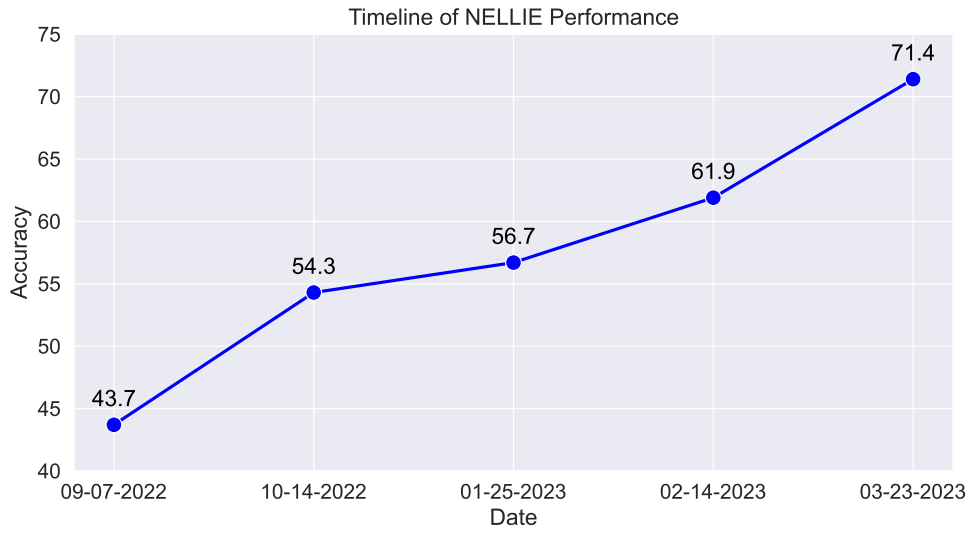


Figure 3.12: Change in NELLIE performance during development period from August 2022 to March 2023.

1. **August 2022**:

   The initial iteration of NELLIE used a T5-Large model trained on 26K decompositions from eQASC and 3.8K decompositions from EntailmentBank. For QA2D declarativization, it used a GPT-J model (Wang and Komatsuzaki, 2021b) performing in-context learning using gold hypotheses from EntailmentBank. Given an input q, a pair, I filled the model's 2048-token context with randomly sampled gold (q_i, a_i, h_i) triplets, then greedily decode

the hypothesis. For filtering 1-premise entailments, it used a Cross-Encoder fine-tuned only on SciTail and for 2-premise entailments it used a (different) T5-Large model fine-tuned positive/negative entailments classification pairs from eQASC and EntailmentBank. We sampled negative examples for EntailmentBank by randomly replacing a child in positive examples with one from the rest of the corpus. For template-conditioned generation, it generated one candidate for each of the 50 most common templates appearing in EntailmentBank (matching 10 or more nodes in the EntailmentBank train split).

2. **September 2022**:

   - Increased the number of generations per template from 1 to 10. This substantially improved the rate at which tempate-conditioned decompositions were used in the search.

3. **October 2022**:

   - Trained an improved cross-encoder classifier for 2-premise entailment. I used a revised strategy for selecting close-but-negative examples from EntailmentBank. I also included the Amazon Mechanical Turk data collected by Tafjord et al. (2022). The resulting model obtained a around .775/.666 precision/recall on a development set containing the gold positive decompositions from EntailmentBank and the negative examples in the Turk data.

- Extended the grounding WorldTree corpus to include the 3K annotator-added facts used as leaves in EntailmentBank trees but not originally in the WorldTree corpus. This substantially raised the proof recall rate for some test hypotheses that were previously ungroundable using just WorldTree. This made the full set of EntailmentBank QA test hypotheses "theoretically groundable."
- Switched the QA2D model from the noisy GPT-J model to a T5-large model trained to map entailment answer options to the gold hypotheses.

4. **December 2022**:

   - Switched the weak unification scoring method from the cos-similarity of the support fact-retrieving bi-encoder (which often is low for even strongly entailing premises) to the confidence of a RTE-trained cross encoder.
   - Added the branch pruning described at the start of Section 3.3.8, which maintains a "current best" score for each hypothesis and prunes all branches that are guaranteed not to find a better score.

5. **February 2023**:

   - Added the 'self-ask' filter described in Section 3.3.7 that checks for model belief in each premise's factuality before accepting it as valid. This was initially implemented using the Entailer-Large model released by Tafjord et al. (2022).

6. **March 2023**:

   During this period, I performed manual error analysis on 150 incorrectly answered questions (75 each from EBQA and WorldTree) and categorized NELLIE's performance by error pattern, yielding the following types:

1. Correct answer outscored by incorrect (29%)	NELLIE found a proof for the correct option, but it scored lower than a proof for an incorrect one.
2. Hypothesis unprovable without the context (23%)	NELLIE's QA2D model produced a question-dependent hypothesis that is not provable without access to the question context.
3. Failed to prove correct answer (19%)	NELLIE failed to find a proof of the correct answer before timing out.
4. True hypothesis for an incorrect option (16%)	the QA2D model produced a hypothesis that is true for an incorrect answer (usually by removing some of the question context)
5. Malformed hypothesis (14%)	The QA2D model produced a correct answer hypothesis that is generally malformed, either by adding or dropping words from the QA pair to introduce ambiguity, or just by producing fragmented language.

Table 3.3: Error class breakdown in March 2023

We observe from this decomposition that around 50% of NELLIE's errors were due to the QA2D model making mistakes (types 2, 4, and 5), not the proof search. The following are an example of error type 4 and 2:

Q: Which statement explains why a mother's unhealthy diet during pregnancy is harmful to her embryo's development?
A: *(wrong)* The embryo inherits half its chromosomes from its mother.

↳ **the embryo inherits half its chromosomes from its mother**

Q: A student placed a limestone chip into a hydrochloric acid solution, and carbon dioxide gas was released. The carbon dioxide was evidence that
A: *(right)* a chemical change occurred.

↳ **a chemical change occurred**

Meanwhile, in the cases that do pertain to the prover (types 1 and 3), 60% were type 1 errors due to an incorrect proof "slipping through the cracks" of NELLIE's verification filters and outscoring a correct answer proof. Because half of all errors were due to declarativization, I pursued the following improvements:

- (Addressing error type 2) Added the capacity to ground to the question context instead of a WorldTree fact.
- (Addressing error types 2,4,5) Replaced the 3 T5-large models serving as the dynamic rule generator, 2-premise entailment filter, and self-ask filter with a single, multi-angle (Tafjord and Clark, 2021) T5-3B model that performs all three functionalities. I fine-tuned this model on all the data used to train the previous iterations. The new T5-3B model and a new 2-premise RTE cross encoder both now conditioned on the question text for all decisions in the algorithm. This addressed challenges faced in the error classes in which declarative hypotheses did not faithfully represent their answer options, as conditioning on the question text allowed the models to consider contextual

statements instead of treating all statements as generics.

Switching to a larger architecture (T5-Large to T5-3B) has implications on the number of candidates one can decode in a single call to Hugging Face's `generate` function for a given checkpoint. I was forced to reduce the number of candidates generated at each recursive step from 100 per call (3 total calls - [DRG with templates, DRG without template, DRG with retrieved facts]) to 40. This coincided to using a stronger generation model in T5-3B that less frequently produced noisy, incoherent decompositions, and so the search still benefited from the switch despite the smaller search horizon.

3.6 Discussion

In this chapter, I proposed a reimagined version of a classical expert system that relies on the inferential power of LLMs rather than handcrafted rules. NELLIE has the skeleton of a symbolic theorem prover, but the provided rules invoke neural predicates that allow for systematic reasoning over NL statements. To dynamically generate inferences, we introduce two mechanisms for knowledge-guided generation: conditioning decompositions on retrieval or model-selected inference templates. Our search algorithm undergirds NELLIE, an explainable reasoning system that performs end-to-end QA by searching for grounded entailment trees. We find that NELLIE equals or exceeds the performance of comparable XQA systems, while providing explanations

with the simultaneous guarantees of logical directionality and groundedness in human-provided text. This work thus suggests a new way to jointly reap the benefits of both modern neural methods and traditional reasoning. It thus is a high-performing embodiment of McCarthy (1959)'s vision of such a system. Finally, we identified a set of promising avenues for future work that builds upon this paradigm, made tractable by NELLIE's modular approach to systematic neuro-symbolic reasoning.

3.6.1 Future Work

We pursue AI solutions that exhibit systematic, interpretable, and granular reasoning behaviors, with NELLIE as an example. NELLIE's modularization of the generation and discrimination of candidates allows us to diagnose paths towards further improvements.

Better Entailment Filters: This work considers mechanisms for improving the *generative* components of our entailment engine: retrieval- and template-conditioned generation as a means to guide search towards factbase-rooted proofs. Future work can consider improved *discriminative* components that are less likely to accept improper entailment hops. Improved filters would also improve search efficiency, as much of NELLIE's budgeted time is spent exploring the branches created by ungroundable subqueries. Beyond using larger and stronger model architectures, improvements to the RTE classifiers might come from constructing domain-relevant training sets containing informative negative examples. Tafjord et al. (2022) suggest one form of

this approach, having their inference model generate proofs of incorrect statements intentionally so that the resulting inference hops can be labelled for acceptability by workers. In Chapter 4, we will explore this avenue in depth, considering a careful protocol for annotating correct and incorrect decompositions generated by the model.

Another avenue for improvement might be through a version of *dataset inoculation* (Liu et al., 2019) or *logic-guided data augmentation* (Asai and Hajishirzi, 2020) stemming from the identification of improper reasoning patterns such as the property inheritance error described above. If these error patterns can be cheaply annotated or instantiated, we can train the classifiers to better identify when to filter them out.

Faster, Smarter Search Strategies: Here we employed breadth-first proof search, making iterative calls to a text generation module that can take multiple seconds. Template-conditioned generation, which improves the quality of the search horizon, takes even longer.[16] Future work can consider more sophisticated methods for candidate selection, so as to make better use of the time budget. The TreeWise search algorithm described in Chapter 4 represents an overhaul of the Nellie logic towards this end.

Making More Questions Groundable: Nellie performs more effectively at QA on the subset of ARC (EntailmentBank) for which there exists a gold explanation tree fully grounded in WorldTree facts, thus guaranteeing that one eixsts. This leads

[16]Imposing our timeout of 180s does not give the system adequate time to explore its entire generated search horizon, as it times out around 7% of the time for correct statements.

to the question: how to increase the coverage of the system's factbase so that more questions can be considered "groundable"? This is part of the motivation for the project discussed in Chapter 6, in which we explore ways to automatically extract discrete, NL textual knowledge representations in order to improve reasoning in new domains.

Noisier Knowledge Sources: The WorldTree knowledge store and EntailmentBank dataset provide a perhaps uniquely tight marriage between a set of questions and the underlying pieces of knowledge needed to answer them; the annotators of WorldTree crafted the datastore specifically to contain only the exact set of facts needed to answer the corresponding questions, no more and no less. Because there is a question related to lions being mammals, the fact "lions are mammals" is in WorldTree, yet "tigers are mammals" is not. WorldTree thus, in a sense, might give away strong hints about the correct answers to questions based on the facts that are present versus those that are not. When I tried NELLIE out on the full ARC-Challenge split (containing mostly questions *not* in either WorldTree or EntailmentBank), performance plummented. This is also reflected by the fact that extending the retrieval index to contain the user-provided facts that appear as leaves in EntailmentBank, we observed a substantial boost in hypothesis grounding and QA performance, and did so again in Figure 3.8 on the OBQA dataset when we added the corresponding knowledge statements. This study thus shows that *if* the knowledge is stored in the corpus, NELLIE can find it, but it can flounder otherwise. But what about a corpus for which we don't know whether the facts are there? Or a corpus that was created completely separately from the QA dataset, thus providing

no guarantees about coverage? In later chapters, I explore using Wikipedia, a general encyclopaedic knowledge source, instead of WorldTree; Wikipedia is full of dense prose containing a wide swath of useful information for ARC and similar datasets, but also a lot of text completely irrelevant to QA. NELLIE's forced-decoding approach to retrieval-conditioned decomposition generation is unlikely to succeed on this corpus, leading me to explore a new instance of the entailment engine framework that has this ability.

Multimodal Knowledge Sources: NELLIE's use of neural models for encoding facts flexibly allows it to reason over NL text rather than symbols, and opens the door to reasoning over a wider variety of knowledge sources than were available to traditional symbolic reasoners. Yet, in this work we have only considered reasoning over a single knowledge source, WorldTree (and other facts from OpenBookQA), and a single medium: text. Yet, neural modeling in other domains such as computer vision and speech processing have shown that neural encoders can represent and reason over these other modalities. In Sanders et al. (2024b), we build upon the NELLIE framework to explore the application of entailment tree search algorithms to multimodal video question answering, combining evidence from both visual and textual information. We achieve state-of-the-art performance on zero-shot VideoQA tasks while providing interpretable reasoning traces. Future work could explore the application of NELLIE to other mixtures of modalities beyond text and video.

Chapter 4

Evaluating Systematic Decompositional Natural Language Inference

Recent language models enable new opportunities for structured reasoning with text, such as the construction of intuitive, proof-like textual entailment trees without relying on brittle formal logic (Tafjord et al., 2022; Weir et al., 2024a). The previous chapter introduced NELLIE, a first instantiation of this idea using one generation of neural sequence and retrieval models. However, progress in this direction has been hampered by a long-standing lack of a clear protocol for determining what *valid compositional entailment* is, which causes noisy datasets and limited performance gains by modern neuro-symbolic engines. To address these problems, this problem makes two contributions. In this chapter we[1] formulate a consistent and theoretically grounded approach to annotating decompositional entailment and evaluating its impact on LLM-based textual inference. We create a new dataset, RDTE (Recognizing Decompositional Textual Entailment), that has a substantially higher internal consistency (+9%) than prior decompositional entailment datasets, suggesting that RDTE is a significant step forward in the long-standing problem of forming a clear protocol for discerning entailment. We then show that RDTE reveals shortcomings in the ability of current neural models to recognize which decompositions humans will consider valid.

[1]This chapter will use the first person plural (we) instead of the singular. This is inspired by tradition (Napoles-Cohen, 2019; Rudinger, 2019; Knowles, 2019; Poliak, 2020) and a desire to recognize the contributions of my collaborators in the project that this chapter describes.

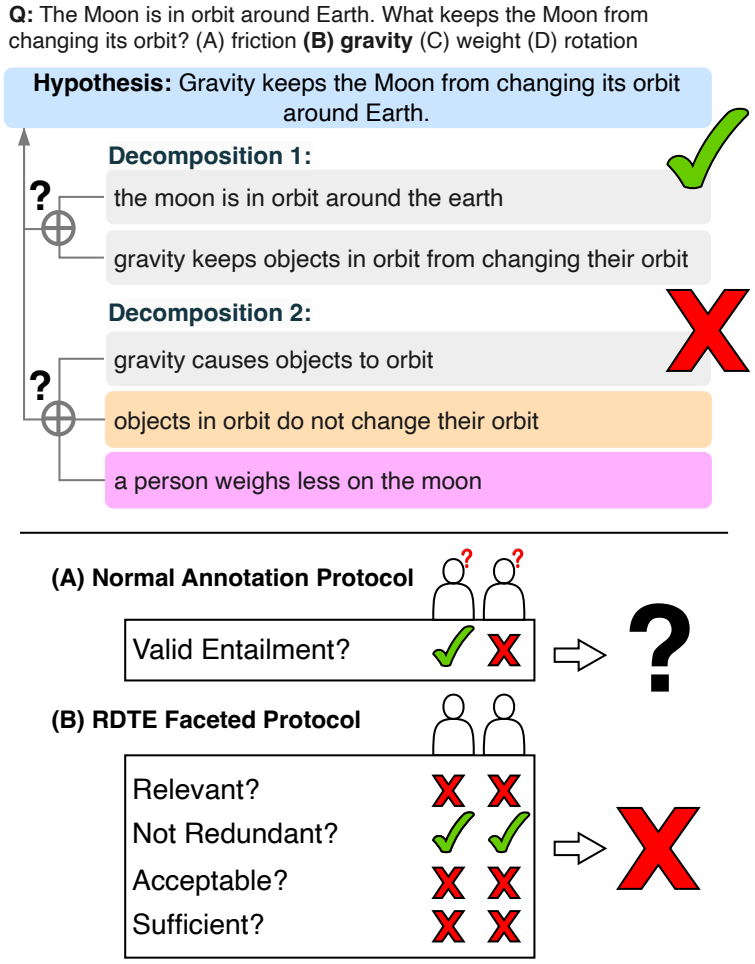


Figure 4.1: **(Upper)** Two hypothesis decompositions suggested by an LLM. The first makes an argument that is generally acceptable to a human. The second contains a fact that is not always true and another that is irrelevant to the entailment. Recognizing such an invalid decomposition is core to recent neuro-symbolic reasoning algorithms, but LLMs struggle at the task. **(Lower)** Ambiguous definitions of entailment have hampered progress in annotating data to improve the models. We find that a faceted definition yields both a clean dataset (RDTE) and significant downstream task improvements.

4.1 Motivation and Overview

Textual inference using entailment trees has emerged as a promising avenue for explainable AI in domains requiring complex reasoning (Dalvi et al., 2021; Bostrom

et al., 2022). Inspired by classical symbolic reasoners that produce proof trees of a hypothesis, recent entailment tree systems (e.g. NELLIE (Weir et al., 2024a) or IRGR (Ribeiro et al., 2022)) explain a hypothesis by constructing, via an LM-based generative search algorithm, a tree of recursive natural language (NL) decompositions whose children entail their parents. Entailment trees provide a natural and granular medium for explaining complex multi-step decisions to users and create opportunities to ground reasoning to large NL knowledge corpora (Weir et al., 2024a).

Entailment tree reasoners are most useful to humans if they are *right for the right reasons*: They not only give correct answers, but each tree step is a valid deduction supporting its hypothesis. While some tree generators leverage classifiers for recognizing textual entailment (RTE; Dagan et al. (2005)) to identify which hypothesis decompositions are illogical and should be discarded (Yang et al., 2022), it is still common for systems to arrive at the correct conclusion while presenting a flawed decomposition, or to accept incorrect conclusions based on faulty logic. Thus, the task of *recognizing decompositional textual entailment* is a major barrier to progress on tree-based reasoners.

Work on entailment trees has largely avoided this critical question of reasoning validity. Existing evaluations focus on tree reconstruction (Ribeiro et al., 2023) and QA performance, avoiding considerations of whether the quality of model-generated trees aligns with the accuracy of the answers. This is largely because reasoning quality is difficult to assess. Evidence suggests humans think of deductive validity

less stringently than formal symbolic axioms; they accept explanations that are more or less incomplete (Sulik et al., 2021; Tan, 2022). Attempts to annotate datasets (Tafjord et al., 2022; Clark et al., 2023) consequently suffer from inconsistent labels, as opinions differ between annotators over what constitutes validity. Without careful instruction, it is hard for annotators to determine where to set their threshold for acceptability on the spectrum from "strict deductive inferences" to "fallible common-sense inferences" (Gubelmann et al., 2023). Thus, asking for a single binary decompositional entailment judgment yields a difficult and noisy annotation task (Figure 4.1, (A)). This leads us to pursue a more consistent protocol for measuring the goodness of one decomposition over another.

The existing discourse surrounding NLI (e.g., Manning (2006)) has not touched on the sort of decompositional entailments encountered in a recursive tree search algorithm. Towards addressing this gap, we propose an evaluation centered on the notion that **a valid decomposition is a valid argument for why the hypothesis should be believed**. We take inspiration from the "Relevance, Acceptability, and Sufficiency" criteria for a logically good argument developed in the field of informal logic (Johnson and Blair, 1977; Groarke, 2022), and design an approach to assessing entailment in a principled manner. We annotate items on ordinal scales targeting different facets of argument analysis, yielding a well-defined and consistent process. We use this protocol to collect over 1000 expert annotations. Through experiments, we find that models trained on previous compositional entailment datasets and LLMs

like GPT fall short of human performance on the dataset, which we term RDTE (Recognizing Decompositional Textual Entailment).

We also find that our annotation protocol can serve as a powerful LM prompting mechanism: Given the same instructions as our human annotators, GPT-4 (OpenAI, 2023) can serve as a "teacher" in a knowledge distillation pipeline that annotates reasoning traces extracted from a tree search algorithm and produces strongly performing student models. For this purpose, we released a large artifact (24K items per domain) of GPT-4's annotations. We illustrate the effectiveness of student models trained via this pipeline as the linchpin of TreeWise, a novel entailment tree engine that performs hypothesis proving via decompositional entailment search. Our contributions are thus:

- **A new entailment challenge set**, RDTE, crafted via an informal logic-inspired protocol, tasking models to validate hypothesis decompositions
- **A knowledge distillation pipeline** that teaches student models to discriminate **what** and **why** decompositions are invalid in a given domain.
- **A new entailment tree-generating inference engine**, TreeWise, which outperforms existing tree-based QA methods while generating higher-quality trees in the process. This engine shows to benefit from RDTE knowledge distillation.

4.2 Decompositional RTE

4.2.1 Task Definition

An entailment tree comprises one or multiple hypothesis decompositions. Each constitutes one instance of decompositional RTE, which we define as follows: Given the question Q, hypothesis h, and premises $[p_1, \ldots, p_n]$, would proving the premises be contextually sufficient to prove h?

As described below, we collect different ordinal scores for each item using criteria from informal logic. Each facet informs the ultimate sufficiency label, for which we evaluate models in §4.5.

4.2.2 Background: RAS Criteria

The field of informal logic (Groarke, 2022), similarly to much of NLI research, develops from the idea that formal deductive logic is unsuitable for the analysis of real-world argumentation. One stated goal is to "develop standards, criteria and procedures for the interpretation, evaluation and construction of arguments and argumentation used in natural language." (Blair and Johnson, 1987). A seminal framework developed by informal logicians to replace strict deductive logic criteria is known as RAS: **Relevance**, **Acceptability**, and **Sufficiency** (Johnson and Blair, 1977). Much like RTE work, RAS assesses whether a hypothesis can be accepted on the basis of a

list of premises. Noting that each element is subject to substantial academic debate (different systems of informal logic define them differently), we review each criterion as we interpret them for this work:

Relevance of premise A concerning conclusion B is defined as whether the truth of A makes a difference to the truth of B. (Blair, 2012). The extent of this difference can vary; e.g. whether (A) "the earth has oxygen" is true technically has relevance to (B) birds can fly (since birds breathe oxygen), but is less relevant than (A') birds have wings. Relevance is of particular interest to a recursive reasoner, as one would not want to waste time proving an irrelevant statement like (A) when proving (B).

Acceptability of a premise is normatively "worthy of acceptance," which can mean either its ostensible truth value or-–in the absence of universal factuality–that in the relevant context, the arguer and the argument recipient accept it to be true. This introduces a hiccup for recursive reasoning algorithms, for which the factuality of a decomposition's premises is commonly determined via search *after* validating the decomposition itself. We ultimately choose to annotate one subset of RDTE for premise factuality while not doing so for the other.

Sufficiency is "the property of an argument's premises of supplying all the grounds that are needed to make it reasonable to believe its conclusion." (Johnson and Blair, 1977). This is left intentionally vague; Blair (2012) notes "the criterion of sufficiency, for justificatory arguments, is best seen as a placeholder for whatever version and standards of sufficiency are appropriate for the particular situation in question." In this

way, we should consider sufficiency to be a question and problem-specific criterion: **the grounds for valid entailment are inherently domain-specific**. Blair (2012) notes the dependent relationship between sufficiency and the other two criteria: sufficient presumes acceptability and relevance.

4.2.3 Implementing RAS for RTE Annotation

We draw inspiration from these criteria to construct a protocol that investigates a decomposition like the ones shown in Figure 4.1 for each RAS component in turn. An important aspect of these normative terms is that they are all scalar: a premise can be more or less relevant and acceptable, and the sufficiency of an argument composed of premises could always be strengthened by adding more premises. In a departure from works such as Tafjord et al. (2022) who collect binary factuality and "reasoning correctness" judgments, we collect RAS judgments on an ordinal scale (Zhang et al., 2017) from 1 to 5, where each score is assigned a specific set of conditions. We also collect an ordinal judgment for **redundancy**, which is not strictly a component of RAS assessment; we observe that redundant premises are particularly problematic for entailment tree search, as proving the same information twice wastes the search budget. We implement redundancy as "conditional irrelevance":

- If removing a premise *in isolation* doesn't change the extent of the entailment, then it's **irrelevant**.

- If removing a premise *in the presence of the other decomposition premises* does not change the extent of the entailment, then it's **redundant**.[2]

The sufficiency label most directly resembles that of a typical RTE label, and is what we evaluate models on in later sections. Following Blair (2012)'s observation, the sufficiency judgment is higly dependent on the other criteria; our annotation directions, found in §4.4, include a substantial list of 30+ conditions around a decomposition that indicate what the sufficiency score should be based on other criteria scores, e.g. "Did you give any fact a 3/5 or lower for redundancy, factuality, or relevance? (Yes = max 4 sufficiency)".

4.3 Data Collection

We seek to construct a dataset of decompositional entailment judgments that are of the kind that a reasoning system might come across while performing QA. Following Tafjord et al. (2022), we found it apt to annotate decompositions generated during a model's reasoning search traces over training questions. We use the backward chaining process of the tree search algorithm TreeWise, which is introduced in §5.2. Each item is a hypothesis and a set of 2-3 premises that the model proposes might conjunctively entail the hypothesis in the context of a given question. To collect a representative sample of hypotheses, we pull from three classes:

[2]An irrelevant premise is thus by definition also redundant.

1. **Top-level correct hypotheses** representing the right answer options for multiple-choice questions. These are important to annotate so that models are *right for the right reasons.*

2. **Recursive correct hypotheses** LLM-generated to decompose the top-level correct hypotheses

3. **Top-level incorrect hypotheses** representing the incorrect answer deemed closest to correct by GPT-4. These are important to annotate so that models are *not wrong for the wrong reasons.*

We collect a mixture of GPT-4 and ChatGPT-generated decompositions using a suite of different styles of prompts described in §5.2.2. We generated decompositions for hypotheses[3] in two task domains: multiple choice science QA in **ARC** (Clark et al., 2018) and multi-hop QA over Wikipedia in **HotpotQA**. While Hotpot questions are constructed over public world knowledge, the esoteric individual facts (e.g. the birth year of some actor) are not common knowledge among humans, and it would be a stretch for the audience of an argument made by a Hotpot decomposition to be expected to know each premise's factuality. We thus do not annotate Hotpot decompositions for factuality, but annotate the other facets as normal.

[3]We created hypotheses by declarativizing (Demszky et al., 2018) answer options using GPT-4. For HotpotQA, we first had GPT-4 synthesize incorrect answer options.

4.3.1 Annotation Process

I and three coauthors (Kate Sanders, Shreya Sharma, Dongwei Jiang), all highly proficient or native English speakers, annotated 1000 decompositions total with two-way redundancy. I annotated all items. We first annotated several examples together to develop a list of 30+ condition checks that indicate certain RAS scores. We then annotated the rest of the dataset independently.

While the dataset is annotated for sufficiency on a 5-point scale, we found a threshold of ≥ 4 fitting to evaluate models under binary entailment metrics. This corresponds to items for which the only acceptable flaws are (A) a 3rd redundant premise in an otherwise perfect 2-premise entailment or (B) minor missing information that does not affect lay reasoning. To arrive at a single clean entailment label for evaluation, we reconciled disagreements by discussing all items for which its two annotated sufficiency scores were on either side of 3.5. The vast majority of disagreements were due to human error, e.g., missing a condition or not recognizing a particular flaw in reasoning.

4.4 RDTE Annotation Instructions

Figure 4.2 shows the rubric and condition lists for evaluating premise-specific facts (relevance, factuality, redundancy). Figure 4.3 shows the rubric and condition list for annotating sufficiency. These are ARC-specific lists; we constructed a similar but

slightly different version for HotpotQA.

Factuality	How factual is the fact? 1 is completely false, 5 is completely true.
Relevancy	How relevant is the fact to helping decompose the conclusion? An irrelevant fact is one that either does not address a key aspect of the conclusion, introduces irrelevant information, or is otherwise unnecessary should be scored lower in relevance. 1 is completely irrelevant, 5 is completely relevant. If the fact is situationally relevant to the conclusion, but contradicts it, you should still score it 5.
Redundancy	Does the fact introduce new information that is not already contained in other facts in the decomposition? 1 is completely redundant, e.g., the third fact completely restates the first one, and 5 is completely new information. Sometimes, one of the facts directly restates the entire conclusion by itself, which **should be marked with the checkmark only and should not affect the numerical score**. Otherwise, facts including information included in the conclusion are acceptable and strictly necessary.

Factuality Questions to Ask Yourself	
Is a fact ambiguously grounded in the question context in a way that does *not* affect the reasoning? (e.g. the fact "two sticks are rubbed together" in the question "what is an example of a force producing heat? (A) two sticks rubbed together, . . . ")	This is acceptable and can be 5/5 factuality
Is a fact true in nearly all cases except extreme ones that don't pertain to the question?	(Yes = 5 factuality)
Relevancy Questions to Ask Yourself	
Is a fact not on topic? ("on topic" is defined as containing nouns or entities that appear in the hypothesis)	(Yes = 1 relevancy)
Does there not exist some (potentially over-pedantic) decomposition in which the given fact would be necessary to complete the entailment?	(Yes = max 2 relevancy)
Would removing an on-topic fact ***in isolation*** **not** change the extent to which the conclusion is supported?	(Yes = 2 relevancy)
Would removing an on-topic fact in isolation **minimally** change the extent to which the conclusion is supported?	(Yes = 3 relevancy)
Redundancy Questions to Ask Yourself	
Is a fact a paraphrase of another fact?	(Yes = 1 redundancy for second fact)
Does a given fact add entailment in isolation, but if you removed the fact ***conditioned on the rest of the facts***, it **would not** change the extent to which the conclusion is supported?	(Yes = max 2 redundancy)
Does a fact restate information in the **question text** (*not* the conclusion)?	This is acceptable. Only check for restatements of other facts and/or the conclusion. Restatement of the question text, especially to cite evidence, is fine.

Figure 4.2: RDTE annotation guidelines for premise-specific qualia in ARC.

1 (Malformed or Nonsensical)	Completely incorrect logic, or contains a fact that contradicts the conclusion, or malformed facts (not complete sentences), or inter-fact pronoun references (e.g. "this" or "that" or "such").
2 (Poor)	Any two of the following: (1) some nontrivial amount of redundancy, (2) one irrelevant fact (2/5 or lower), (3) missing/implicit information that makes deducing the conclusion impossible without a substantial leap in logic. Would not convince a human of the conclusion.
3 (Moderately Correct)	Generally coherent and correct, but there is some significant flaw. E.g., **one of the facts is untrue** but if it was true the proof would be correct.
4 (Mostly Correct)	Slight redundancy or missing/implicit information, but **not to the point that it should substantially impact a human** performing the reasoning.
5 (Perfect)	Completely correct and sound decomposition. No redundancy and no missing/implicit information. No ambiguous language. Addresses all conditions required to infer the conclusion. Does not leave anything implicit.

Questions to Ask Yourself	
Are any of the premises not well-formed? (no fragments, no 1 sentence that was split into two non-sentence parts)	(Yes = 1)
Do the premises together or individually *contradict* the conclusion instead of supporting it?	(Yes = 1)
Are there between-premise pronoun references ('this', 'that', 'such') whose antecedent would be ambiguous without the other premises?	(Yes = 1)
Are there any conjunctive adverbs like "therefore" or "thus"?	(Yes = 1)
Are all premises irrelevant, off-topic, or not contributing any correct logic? (e.g. all 1's for relevance, or removing all of them would not change the extent of the entailment)	(Yes = 1)
Does any premise essentially restate the conclusion without adding/removing any information?	(Yes = max 2)
Are there at least two of the following? (1) redundant fact, (2) untrue fact, (3) irrelevant fact, (4) missing information	(Yes = max 2)
Is the conclusion assuming an effect that isn't directly linked to the cause, or overlooking more immediate effects?	(Yes = max 2)
If you removed all redundant or irrelevant (3/5 or lower) facts, would there be only one fact remaining **and not full entailment**?	(Yes = max 2)
Is there any amount of logical error present?	(Yes = max 2)
Did you give any fact a 2/5 or lower for factuality or relevance?	(Yes = max 3)
Does proving one premise amount to proving all of the others?	(Yes = max 3)
Are the premises all factual and relevant, but there is a part of the conclusion (e.g. something non-common-sense or a thing that a 10-year-old would **not** intuit in the context of the question) that is not stated or explained?	(Yes = max 3)
If you removed all redundant or irrelevant (3/5 or lower) facts, would there be only one fact remaining?	(Yes = max 3)
Are the premises all evidence statements entailed by the question context and nothing else?	(Yes = **likely** max 3)
Is one of the facts not true, but if it were then it'd be a perfect entailment?	(Yes = 3)

Figure 4.3: RDTE annotation guidelines for annotating sufficiency in ARC (continued on next page).

Questions to Ask Yourself	
Did you give any fact a 3/5 or lower for redundancy, factuality, or relevance?	(Yes = max 4)
Are there two separate arguments being partially/mostly made to support the hypothesis, but one/both is missing some implicit premises?	(Yes = max 4)
Are the premises all factual and relevant, but the conclusion does not follow from the premises for a minor reason (e.g. a common sense-y fact that would have been inferred by a 10-year-old in the context of the question)	(Yes = max 4)
Do two of the facts perfectly entail the conclusion, but the third is essentially redundant?	(Yes = 4)
Is the conclusion irrelevant to the question, but the entailment supports the conclusion perfectly?	(Yes = 5)
Does the question ask something along the lines of "which is the best. . . ?" and the entailment doesn't mention the other answer options?	Treat it like the "best" is not there
Do the premises not properly entail the conclusion for some other reason?	Reach out to us for clarification
Is one premise P1 an effective paraphrase of the hypothesis H, but another premise P2 serves as lexical grounding between P1 and H?	Ignore the paraphrase = max 2 rule

Figure 4.4: (Continued) RDTE annotation guidelines for annotating sufficiency in ARC.

4.4.0.1 Comparison to Existing Data

Prior to reconciling disagreements, we measured raw annotation agreement and compared it to recent attempts to collect decompositional entailment labels. Tafjord et al. (2022) collected annotations for 3.7K decompositions generated by **Entailer**, while Clark et al. (2023) did so for 24.4K items generated by **GPT3**.[4] The instructions given to annotators for these datasets are very high-level, simply asking whether the "reasoning goes wrong." We believe this creates a much lower threshold for acceptability than the RDTE protocol; these datasets are majority labeled "entailment," but on manual inspection, we found numerous items that exhibited redundant, irrelevant, and fallacious arguments.

[4]The BaRDa dataset comprises a specifically filtered subset of these two datasets that exhibit maximal annotator agreement. We obtained the pre-filtered data from the authors for the purposes of comparison with our work.

Nevertheless, RDTE has a higher internal annotator consistency rate than these datasets, with a raw agreement rate 9% higher and Krippendorf α 19 points higher than the previous best.

	RDTE (ours)	**Entailer**	**GPT3**
Raw agreement	**.79**	.70	.61
Krippendorf α	**.53**	.34	.31

Table 4.1: Agreement statistics in RDTE compared to existing decompositional RTE datasets.

4.4.1 RDTE Analysis

Figure 4.5 depicts the breakdown of sufficiency labels within the RDTE ARC (267) and Hotpot (775) decompositions. While the ordinal scores are relatively well distributed, only 27% of the dataset is labeled a 4 or 5, creating a large binary label imbalance. This statistic highlights the importance of performing well on this task: 3 out of every 4 decompositions generated by ChatGPT and GPT-4 did not pass our sufficiency rubric.

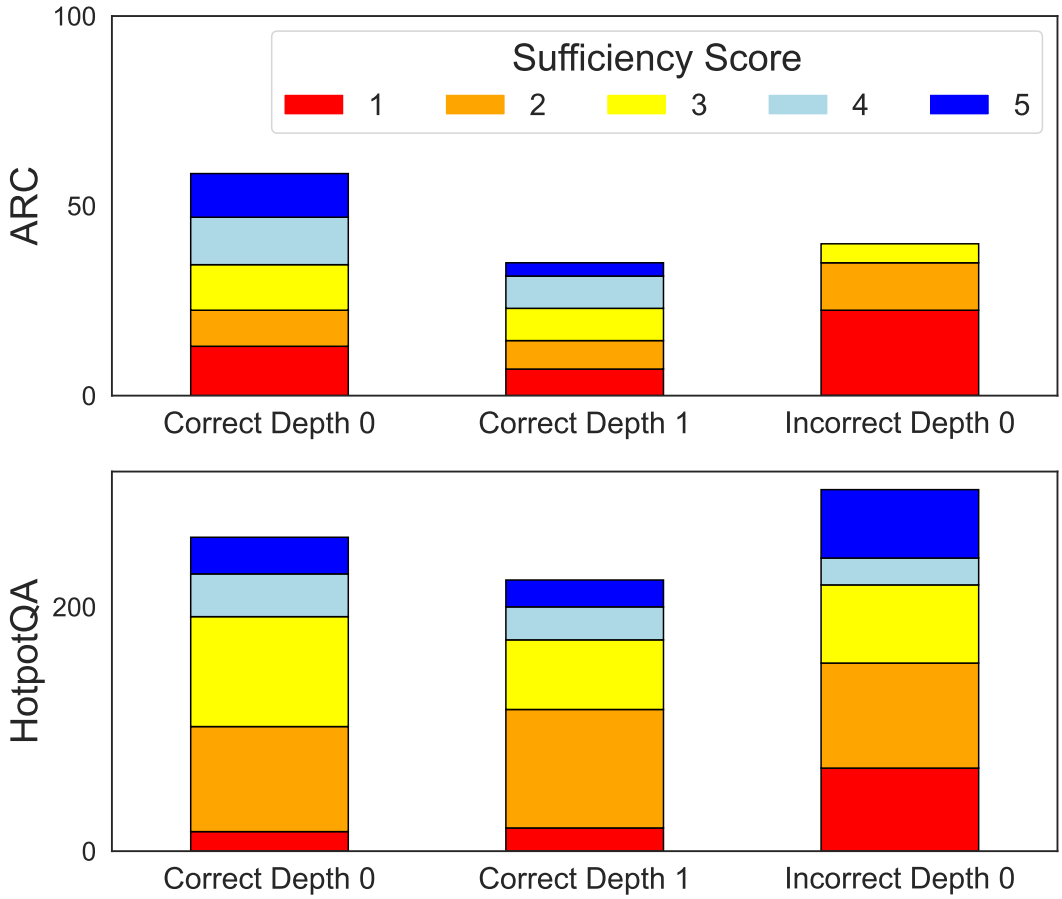


Figure 4.5: Distribution of the 1000 entailment labels in RDTE. Instead of binary entail/non-entailment, we annotate on a 5-point ordinal scale. To evaluate binary judgment models, we treat $\geq$4 as positively labeled.

49% of RDTE items had at least one premise rated as irrelevant ($\leq$ 3/5); 36% had at least one rated as redundant. 28% of the items with a factuality rating had at least one nonfactual premise. We note that *none of the incorrect ARC hypothesis decompositions were labeled as sufficient*; this highlights that the RDTE protocol, when including the factuality facet, does a comprehensive job of systematically ruling out decompositions for which there is necessarily some issue (else the hypothesis must be correct). Contrastly, there *are* positively-labeled decompositions of incorrect

hypothesis for Hotpot, because we did not annotate for factuality. There are thus decompositions that, were their premises true, would entail a wrong hypothesis.

	Fa	**Rel**	**Red**
Q: The 2005 film Remedy featured Frank Q: Vincent from The Sopranos and several mob movies by which acclaimed director? (A) Jonathan Demme, (B) Martin Scorsese, (C) Ralph De Vito, (D) Steven Hilliard Stern **H**: The 2005 film Remedy featured Frank Vincent from The Sopranos and several mob movies by the acclaimed director Ralph De Vito.			
P1: Frank Vincent appeared in The 2005 film Remedy.	–	5	5
P2: Frank Vincent appeared in The Sopranos and several mob movies	–	5	5
P3: Ralph De Vito directed several mob movies.	–	5	5
Sufficiency: 3. The facts are relevant and nonredundant, but they do not link the mob movies by De Vito to those that featured Vincent. Substantial missing information is a 3.			
Q: Which process best explains how the Grand Canyon became so wide? (A) folding, (B) erosion, (C) deposition, (D) sedimentation **H**: Erosion best explains how the Grand Canyon became so wide.			
P1: Changes in the Grand Canyon's landscape include becoming wider.	5	5	5
P2: Erosion is one of the processes that can change a landscape.	5	5	5
P3: The Grand Canyon is a landscape.	5	3	5
Sufficiency: 2. The facts are relevant and nonredundant, but they do not establish a direct causal relationship between erosion and the Grand Canyon becoming wider. Removing P3 does not strongly impact the extent of the entailment. Redundant P + missing information is a 2.			
Q: Is Ordos City more west than Yangzhong? (A) No, (B) Same longitude, (C) yes **H**: Ordos City is located in the western part of China.			
P1: Ordos City is predominantly rural.	-	3	5
P2: Predominantly rural areas in China are often found in the western part of the country.	-	3	5
Sufficiency: 2. The argument commits a fallacy of division by assuming that because predominantly rural areas in China are often in the west, Ordos City, being rural, must also be in the western part. This reasoning does not adequately support the conclusion without specific geographic evidence. Fallacious reasoning and irrelevant premises is a 2.			

Figure 4.6: Example RDTE annotations.

4.5 RDTE Evaluation

We elicit sufficiency judgments over the 1000 gold-annotated RDTE items using a series of methods based on the RDTE protocol and existing methods for judging decompositional entailment. We test them on the gold sufficiency scores converted to binary labels by assiging 4 and 5s to positive entailment. Factuality, relevance, and redundancy predictions were not directly evaluated. We turn off this functionality for the Hotpot subset of RDTE for methods that provide factuality judgments as part of their inference process (this entails either rewriting a prompt or not making a second model call).

4.5.0.1 GPT Methods

We test the following prompt-based methods, using both GPT-4 and ChatGPT.[5] All prompts can be found in the appendix.

- An **ICL** prompt containing the RDTE annotation rubric and 4 example batches (10-15 decompositions per hypothesis) and a **Zero-Shot** prompt containing only the RDTE rubric.
- The prompts used by Clark et al. (2023) to evaluate models on the **BaRDa** dataset. These are separate prompts for the entailment and premise factuality judgments (see Figure 4.10).

[5] `gpt-4-0613` and `gpt-3.5-turbo-1106`

4.5.0.2 Existing Methods

We test various RTE models from recent entailment tree-based reasoners: the T5-based **Entailer-11B** (Tafjord et al., 2022) and the similar **NELLIE-3B** model (Weir et al., 2024a), as well as the fine-tuned RoBERTa classifier used by the NELLIE system as an entailment filter. We also take an off-the-shelf (**OTS**) RoBERTa finetuned for NLI using a suite of 600 datasets[6] that includes QASC (Khot et al., 2020) and ARC.

4.5.0.3 Knowledge Distillation on Silver Data

Search algorithms like TREEWISE make hundreds of calls to NLI models for every question, making large, slow models like GPT-4 unrealistic for use as an entailment filter. We instead explore using knowledge distillation to train smaller student models to imitate GPT-4's performance. We extracted 24K RDTE judgments from GPT-4 over 2.3K hypotheses in each of the ARC and Hotpot domains. We release these large silver datasets for future work.[7]

We used the silver data to fine-tune (A) the RoBERTa model used in the NELLIE engine, and (B) ChatGPT using OpenAI's fine-tuning API. We trained the former to predict the binarized silver sufficiency judgments and the latter to replicate GPT-4's exact answers to the RDTE prompt. For token efficiency, the GPT-4 teacher and ChatGPT student perform classification over a batch of decompositions for a

[6] huggingface.co/sileod/deberta-v3-large-tasksource-nli

[7] GPT-4 obtains a .61 and .44 Spearman ρ with humans on the ARC and HotpotQA RDTE splits, respectively.

single hypothesis in each prompt.[8] We train the ChatGPT student for 5 epochs using the OpenAI fine-tuning API, and the RoBERTa student for 10 epochs using the SentenceTransformers library (Reimers and Gurevych, 2019). We trained separate student models on 18.3K ARC decompositions and on 20.4K Hotpot ones.

RDTE judgment prompt:

```
You are a reasoning system that searches for proofs of a hypothesis by recursively decomposing it into simpler premises.

Given a question and a hypothesis, you give a list of possible decompositions of the hypothesis into premises such that proving the list
      of premises would amount to proving the hypothesis through compositional entailment. The hypothesis might represent an answer to
      the question ,or it might represent a recursive query. However, many of the decompositions are incorrect, and you must identify
      which ones are correct and which ones are incorrect. For the following question, hypothesis and list of premise decompositions,
      score each decomposition according to the following rubrics:

You will first score each premise on a scale of 1 to 5 for each of the following qualia:
Factuality: How factual is the premise? 1 is completely false, 5 is completely true.
Relevance: How relevant is the premise to helping explain the hypothesis? 1 is completely irrelevant, 5 is completely relevant.
Redundancy: Does the premise introduce new information that is not already contained in other premises? 1 is completely redundant, i.e.
      completely restating another premise or the hypothesis, 5 is completely new information.

You will then judge whether the decomposition as a whole constitutes a complete inference, on a scale of 1 to 5 using the following
      rubric:
1 (malformed or nonsensical): Completely incorrect logic, or contains a premise that contradicts the hypothesis, or malformed instances,
      or inter-premise pronoun references (E.g. a "this" in premise 2 that refers to premise 1).)
2 (poor): Some nontrivial amount of redundancy, one irrelevant fact, and/or missing information. Would impact a human performing
      reasoning.
3 (moderately correct): Generally coherent and correct, but there is some significant flaw. (E.g., one of the facts is untrue but if it
      was true the proof would be correct.)
4 (mostly correct): Slight redundancy or missing information, but not to the point that it should substantially impact a human performing
       the reasoning.
5 (perfect): Completely correct and sound decomposition. No redundancy and no missing information. No ambiguous language.

You are renowned for your stringent eye. There should be minimal "information loss" between the hypothesis and the premises; you are
      looking for strict entailment. YOU RARELY GIVE A 5

Finally, you will provide an explanation for your judgment. Your explanation should justify any non-perfect scores you have given for
      factuality, relevance, and redundancy. In other words, explain why you gave a premise a certain score based on the information in
      the premise and its relation to the hypothesis.
For the complete inference score, explain why the conjunction of premises either does or does not amount to a complete and correct proof
      of the hypothesis. If there were issues with the complete inference, identify what information was missing or what logical errors
      were made. Your explanation should be clear and concise, providing valuable insight into your scoring process

Your output should be serialized json items, one per line, and nothing else.
```

Figure 4.7: Directions used to extract RDTE judgments from instruction-tuned models. The zero-shot prompt used for knowledge distillation contains these instructions but no in-context learning exemplars, which are shown in the following pages and used by methods Table 4.2 that are followed by "(ICL)".

[8] Batching barely affects performance on RDTE (Table 4.2).

RDTE judgment prompt (continued):

```
QUESTION 1:
Which of the following items conducts electricity? (A) a lego brick, (B) a suit of armor, (C) a wooden table, (D) a T-shirt

HYPOTHESIS 1:
A suit of armor conducts electricity

DECOMPOSITIONS 1:
(1) a suit of armor is made of iron AND iron is a metal
(2) armor is made of metal AND metal conducts electricity
(3) armor cannot be punctured AND iron conducts electricity
(4) armor is made of iron AND iron is a metal AND metal conducts electricity
(5) armor is an object AND objects conduct electricity AND armor is an object that is made of metal
(6) a wooden table is made of wood AND wood conducts electricity
(7) conductivity is the degree to which a material conducts electricity AND conductivity is measured as the ratio of current density to
      the electric field that causes the flow of current.
(8) an item conducts electricity if the material that it is made of conducts electricity AND metal conducts electricity
(9) a suit of armor is made of iron AND iron is a metal AND conductivity is measured in Siemens per meter
(10) metal conducts electricity AND a suit of armor conducts electricity

JUDGEMENTS 1 (10 items):
{{"index": 1, "factuality": [4, 5], "relevance": [5, 5], "redundancy": [5, 5], "complete_inference": 2, "explanation": "The fact that
      armor is made of iron and iron is a metal does not necessarily mean that armor conducts electricity."}}
{{"index": 2, "factuality": [5, 5], "relevance": [5, 5], "redundancy": [5, 5], "complete_inference": 5, "explanation": "Properly
      identifies that armor is made of a type of material (metal) that conducts electricity."}}
{{"index": 3, "factuality": [5, 5], "relevance": [1, 5], "redundancy": [5, 5], "complete_inference": 2, "explanation": "The fact that
      armor cannot be punctured is irrelevant to whether armor conducts electricity."}}
{{"index": 4, "factuality": [3, 5, 5], "relevance": [5, 5, 5], "redundancy": [5, 5, 5], "complete_inference": 4, "explanation": "
      identifies that armor is made of a material (iron) that is a type (metal) that conducts electricity, but armor is not always made
      of iron."}}
{{"index": 5, "factuality": [5, 2, 5], "relevance": [3, 3, 5], "redundancy": [1, 5, 5], "complete_inference": 3, "explanation": "Not all
      objects conduct electricity, so the premise 'objects conduct electricity' is an overgeneralization."}}
{{"index": 6, "factuality": [5, 1], "relevance": [1, 2], "redundancy": [5, 5], "complete_inference": 2, "explanation": "The premises are
      about a wooden table, not referencing a suit of armor, and wood does not conduct electricity."}}
{{"index": 7, "factuality": [5, 5], "relevance": [1, 1], "redundancy": [5, 5], "complete_inference": 2, "explanation": "The premises are
      about the general concept of conductivity, not specifically about a suit of armor or metal."}}
{{"index": 8, "factuality": [5, 5], "relevance": [5, 5], "redundancy": [5, 5], "complete_inference": 3, "explanation": "Does not include
      that a suit of armor is made of metal"}}
{{"index": 9, "factuality": [5, 5, 5], "relevance": [5, 5, 1], "redundancy": [5, 5, 5], "complete_inference": 2, "explanation": "the
      measurement of conductivity in Siemens per meter is unnecessary to prove that a suit of armor conducts electricity."}}
{{"index": 10, "factuality": [5, 5], "relevance": [5, 5], "redundancy": [5, 1], "complete_inference": 2, "explanation": "Second premise
      restates the hypothesis"}}

QUESTION 2:
Natural selection cannot occur without (A) competition for unlimited resources. (B) gradual warming of Earth. (C) genetic variation in
      species. (D) asexual reproduction in species.

HYPOTHESIS 2:
Natural selection cannot occur without competition for unlimited resources.

DECOMPOSITIONS 2:
(1) Competition for resources can lead to individuals with favorable traits reproducing more AND Natural selection is the process where
      organisms with favorable traits are more likely to reproduce
(2) Competition is a natural occurrence when resources are limited AND Unlimited resources can lead to an absence of competition AND
      Without competition, there's no natural selection
(3) The limited availability of a required resource can make survival and growth difficult for a species AND if an organism survives
      competition for resources then this organism will survive / reproduce for natural selection
(4) competition for resources is a selection pressure AND in nature, resources are limited AND natural selection occurs when there is
      competition for resources
(5) competition for unlimited resources is necessary for natural selection to occur AND limited availability of a required resource can
      make survival and growth difficult for a species
(6) natural selection requires competition for unlimited resources AND the survival of species depends on available resources and lack of
       competition
(7) competition for unlimited resources is a form of competition that drives natural selection AND natural selection is the process by
      which organisms with beneficial traits survive competition to reproduce and pass on those traits
(8) The survival of species depends on available resources and lack of competition AND limited availability of water can be a selection
      pressure leading to evolutionary adaptations AND without photosynthesis, a plant cannot produce the food it needs to grow and
      survive
(9) Competition for unlimited resources could be treated as a driving pressure for evolution AND Natural selection is one of the
      mechanisms of evolution
(10) competition for resources is a form of competition that drives natural selection AND natural selection is the process by which
      organisms with beneficial traits survive competition to reproduce and pass on those traits
(11) competition for unlimited resources provides a greater opportunity for organisms with beneficial traits to have an advantage in
      survival and reproduction AND natural selection requires competition for resources in order for organisms with beneficial traits to
       have an advantage in survival and reproduction
(12) The survival of species depends on available resources and lack of competition AND in nature, resources are limited
```

Figure 4.8: In-context learning prompt for using the RDTE protocol (2/3)

RDTE judgment prompt (continued):

```
JUDGEMENTS 2 (12 items):
{{"index": 1, "factuality": [5, 5], "relevance": [5, 5], "redundancy": [5, 5], "complete_inference": 3, "explanation": "The decomposition
       does not specify the role of 'unlimited resources' in the competition."}}
{{"index": 2, "factuality": [5, 5, 5], "relevance": [5, 1, 5], "redundancy": [5, 5, 5], "complete_inference": 2, "explanation": "The
      second premise weakens the hypotheses, the first premise is irrelevant, and the last premise doesn't address unlimited resources"}}
{{"index": 3, "factuality": [5, 5], "relevance": [5, 5], "redundancy": [5, 5], "complete_inference": 2, "explanation": "The decomposition
       does not specify the role of 'unlimited resources' in the competition."}}
{{"index": 4, "factuality": [5, 5, 5], "relevance": [5, 2, 5], "redundancy": [5, 5, 5], "complete_inference": 2, "explanation": "The
      decomposition does not specify the role of 'unlimited resources' in the competition, and premise 2 is not very relevant"}}
{{"index": 5, "factuality": [5, 5], "relevance": [5, 5], "redundancy": [1, 5], "complete_inference": 2, "explanation": "the first premise
       restates the hypothesis"}}
{{"index": 6, "factuality": [5, 1], "relevance": [5, 5], "redundancy": [5, 5], "complete_inference": 2, "explanation": "The second
      premise is untrue, the survival of species does not depend on a lack of competition."}}
{{"index": 7, "factuality": [1, 5], "relevance": [5, 5], "redundancy": [5, 5], "complete_inference": 3, "explanation": "premise 1 is
      false."}}
{{"index": 8, "factuality": [1, 5, 5], "relevance": [1, 1, 1], "redundancy": [5, 5, 5], "complete_inference": 1, "explanation": "The
      premises are about water and photosynthesis, not competition for unlimited resources."}}
{{"index": 9, "factuality": [2, 5], "relevance": [5, 5], "redundancy": [5, 5], "complete_inference": 2, "explanation": "premise 1 is
      effectively untrue, and the decomposition does not explain how natural selection requires the competition for resources."}}
{{"index": 10, "factuality": [5, 5], "relevance": [5, 5], "redundancy": [3, 5], "complete_inference": 4, "explanation": "Premise 1 is
      somewhat redundant given premise 2, and the decomposition does not clearly explain why unlimited resources specifically are
      necessary for natural selection to occur."}}
{{"index": 11, "factuality": [1, 5], "relevance": [5, 5], "redundancy": [5, 5], "complete_inference": 3, "explanation": "premise 1 is
      untrue."}}
{{"index": 12, "factuality": [1, 5], "relevance": [1, 1], "redundancy": [5, 5], "complete_inference": 2, "explanation": "The first
      premise is untrue, the survival of species does not depend on a lack of competition, and the second is irrelevant to the question
      of unlimited resources"}}

QUESTION 3:
The gravitational force between the Moon and Earth depends on (A) their masses, only, (B) their diameters, only, (C) their masses and how
       far apart they are, (D) their diameters and how far apart they are

HYPOTHESIS 3 (RECURSIVE):
the Moon and Earth are two objects

DECOMPOSITIONS 3:
(1) the Moon and Earth are at a certain distance apart AND the Moon and Earth have masses
(2) Earth is an object AND planets are objects AND the Moon is an object
(3) Earth is an object AND natural satellites are objects AND the Moon is a natural satellite
(4) the Moon and Earth have diameters AND the Moon and Earth have masses
(5) the Moon and Earth are at a certain distance apart AND the Moon and Earth have diameters
(6) all existing entities in space are regarded as objects AND the Earth exists AND the Moon exists
(7) the Earth is an object AND the Moon is an object AND two objects can exert gravitational force on each other
(8) Earth is an object AND the Moon is an object
(9) objects with mass are considered as objects in physics AND the Moon and Earth both have mass AND the Earth has mass
(10) the Earth is an object AND the Moon is an object

JUDGEMENTS 3 (10 items):
{{"index": 1, "factuality": [5, 5], "relevance": [1, 5], "redundancy": [5, 5], "complete_inference": 2, "explanation": "premise 1 is
      irrelevant to the hypothesis"}}
{{"index": 2, "factuality": [5, 5, 5], "relevance": [5, 5, 5], "redundancy": [5, 2, 5], "complete_inference": 4, "explanation": "that
      planets are objects is unnecessary given premise 1"}}
{{"index": 3, "factuality": [5, 5, 5], "relevance": [5, 5, 5], "redundancy": [5, 5, 5], "complete_inference": 5, "explanation": "The
      premises correctly entail that the Earth and the Moon (a natural satellite) are both objects."}}
{{"index": 4, "factuality": [5, 5], "relevance": [1, 5], "redundancy": [5, 5], "complete_inference": 2, "explanation": "premise 1 is
      irrelevant"}}
{{"index": 5, "factuality": [5, 5], "relevance": [1, 1], "redundancy": [5, 5], "complete_inference": 1, "explanation": "both premises are
       irrelevant to the hypothesis"}}
{{"index": 6, "factuality": [5, 5, 5], "relevance": [5, 5, 5], "redundancy": [5, 5, 5], "complete_inference": 5, "explanation": "
      correctly identifies that the Earth and the Moon, which both exist, are considered objects in space."}}
{{"index": 7, "factuality": [5, 5, 5], "relevance": [5, 5, 1], "redundancy": [5, 5, 5], "complete_inference": 3, "explanation": "the
      third premise about gravitational force is not necessary to prove the hypothesis."}}
{{"index": 8, "factuality": [5, 5], "relevance": [5, 5], "redundancy": [5, 5], "complete_inference": 5, "explanation": "The premises
      properly entail that both things are objects."}}
{{"index": 9, "factuality": [5, 5, 5], "relevance": [5, 5, 5], "redundancy": [5, 5, 1], "complete_inference": 3, "explanation": "Premise
      3 is redundant given premise 2"}}
{{"index": 10, "factuality": [5, 5], "relevance": [5, 5], "redundancy": [5, 5], "complete_inference": 5, "explanation": "directly states
      that both the Earth and the Moon are objects."}}

QUESTION 4:
{question}

HYPOTHESIS 4 {recursive_or_not}:
{hypothesis}

DECOMPOSITIONS 4:
{decompositions}

JUDGEMENTS 4 ({n_items} items):
```

Figure 4.9: In-context learning prompt for using the RDTE protocol (3/3)

```
In the following exercise, I would like you to tell me if a line of reasoning is reasonable or not.

I will give you some facts and a possible conclusion. Please tell me whether the conclusion reasonably follows from the facts I gave you.
       If the conclusion does reasonably follow from the facts, then please answer "yes". If the conclusion does not reasonably follow
      from the facts, then please answer "no".

Note that some of the facts may be false, but I am only interested whether the conclusion would reasonably follow IF those facts were
      true. In other words, imagine a world in which the given facts are true. Would it be reasonable to draw the conclusion from those
      facts, if they were true?

Here are some examples:

IF Vegetables are plants.
AND Cabbages are plants.
THEN Cabbages are vegetables.
Q: Does the rule's conclusion reasonably follow from the facts in the condition, if they were true? A: no

IF a nail is made of metal
AND metals conduct electricity
THEN a nail conducts electricity.
Q: Does the rule's conclusion reasonably follow from the facts in the condition, if they were true? A: yes

IF dogs are birds
AND birds can fly
THEN dogs can fly
Q: Does the rule's conclusion reasonably follow from the facts in the condition, if they were true? A: yes

IF sound requires matter to travel
AND a vacuum has no matter in it
THEN sound will not travel in a vacuum.
Q: Does the rule's conclusion reasonably follow from the facts in the condition, if they were true? A: yes

IF Erosion can cause a landslide.
AND Mud is deposited by a landslide.
THEN Erosion can cause mud to be deposited.
Q: Does the rule's conclusion reasonably follow from the facts in the condition, if they were true? A: yes

IF An animal needs to breathe in order to live.
AND Living things need water to live.
THEN Animals need water to live.
Q: Does the rule's conclusion reasonably follow from the facts in the condition, if they were true? A: yes

IF Frogs also have a larynx, or voice box, to make sounds.
AND Animals that have vocal cords can make sounds.
THEN Frogs are animals.
Q: Does the rule's conclusion reasonably follow from the facts in the condition, if they were true? A: no

IF All humans breathe.
AND Stones breathe.
THEN All humans and stones breathe.
Q: Does the rule's conclusion reasonably follow from the facts in the condition, if they were true? A: yes

IF If a planet is rocky, it can only have a thin atmosphere.
AND Small planets and rocky planets have very thin atmospheres.
THEN If a planet is small and rocky, it has a thin atmosphere.
Q: Does the rule's conclusion reasonably follow from the facts in the condition, if they were true? A: yes

IF Damming a river can cause a lake to form.
AND Dams are made of concrete.
THEN Dams are concrete lakes.
Q: Does the rule's conclusion reasonably follow from the facts in the condition, if they were true? A: no

Now your turn!
{Entailment}
```

```
Here are some examples:
Is it true that an ocean contains large bodies of water? yes
Is it true that lightning is similar to a volcano erupting? no
Is it true that a fox squirrel is a kind of animal? yes
Is it true that a rainbow is a kind of electromagnetic discharge? no
Is it true that the surface of the moon is made up of water? no
Is it true that the surface of the moon is made up of gases? no
Is it true that a bluebird is a kind of animal? yes
Is it true that the moon 's surface is made up of oceans? no
Is it true that the opposite of negative impact is positive impact? yes
Is it true that building a new highway through the area has a negative impact on the ecosystem? yes

Now let's do some more! Remember, answer with just a single word, yes or no.
Is it true that} \normalsize {\it insert the statement to assess here}
```

Figure 4.10: BaRDa prompts from Clark et al. (2023) for entailment (upper) and factuality (lower) judgments.

4.5.1 RDTE Results

Our main metric is **F-score ($\beta = 0.5$)**, putting double precedence on precision over recall. Precision is particularly crucial for our needs: false positives in a backward-chaining search can quickly create error propagation, wasted time on invalid search branches, and worse trees. In contrast, while recall is important, it is less catastrophic to overfilter valid decompositions than to underfilter bad ones.

Table 4.2 shows $F_{0.5}$ performance by models on the RDTE dataset. We also display precision and recall. We observe that **no model cracks 70**, suggesting room for future improvement. Overall we find the **RDTE-oriented prompts to GPT-4 outperform all existing methods for compositional entailment** by 10 or more points.[9]

We find that knowledge distillation proves effective on RDTE: the student ChatGPT models improve 5-13% over the closest ChatGPT methods, while the **fine-tuned RoBERTa student ultimately outperforms the teacher GPT-4 method**. Table 4.2 shows that this is due to higher precision (68 to 55) at the expense of recall (57 to 90). The RoBERTa student is the most precise classifier tested, and the ChatGPT student is the next-highest among non-GPT-4 models; however, both show much lower recall than the others tested.

[9] **??** shows GPT-4's score when thresholding on **5**, not **4**, which we found to score lower.

		RDTE–ARC			RDTE–Hotpot			BaRDa		
	Training Data	**Pr**	**Re**	**$F_{0.5}$**	**Pr**	**Re**	**$F_{0.5}$**	**Pr**	**Re**	**$F_{0.5}$**
Itemwise GPT Methods										
GPT-4 (ICL)	RDTE Directions + 4 Exemplars	55	90	**59**	**49**	74	**53**	**92**	46	76
(w/ Threshold 4)		49	**100**	55	45	86	50	90	58	81
GPT-4 (Zero-Shot)	RDTE Directions Only	59	57	58	47	60	49	91	31	66
(w/ Threshold 4)		39	**100**	45	35	91	40	83	70	80
GPT-4 (BaRDa)	BaRDa Directions + 10 Exemplars	40	72	44	43	95	48	82	85	**83**
ChatGPT (ICL)	RDTE Directions + 4 Exemplars	31	97	36	30	95	35	69	91	72
ChatGPT (Zero-Shot)	RDTE Directions Only	36	64	40	39	33	38†	72	15	41
ChatGPT (BaRDa)	BaRDa Directions + 10 Exemplars	38	94	43†	30	79	34	73	84	75
Batched GPT Methods										
GPT-4 (ICL)		52	79	56	52	62	54		(N/A)	
GPT-4 (Zero-Shot)		52	64	54	52	51	52		(N/A)	
T5 and Cross Encoders										
Entailer-11B	EntailmentBank + Entailer	45	68	48	33	90	38	76	83	77
NELLIE-3B	EntailmentBank + Entailer + QASC	39	74	43	31	95	36	72	94	75
NELLIE RoBERTa Filter	EntailmentBank + Entailer + QASC	33	76	37	32	95	36	68	**95**	72
OTS NLI RoBERTa	Various NLI incl QASC	42	68	45‡	39	85	44‡	76	90	79
Knowledge Distillation Students										
ChatGPT	GPT-4 (Zero-Shot) Silver Data	46	58	48†	52	49	51†	82	55	75
RoBERTa	GPT-4 (Zero-Shot) Silver Data	**68**	57	**66**	**56**‡	56	**56**‡	83	47	72

Table 4.2: Entailment results ($F_{0.5}$ score). by various approaches on the RDTE and BaRDa datasets. We find that RDTE prompting of GPT-4 greatly outperforms existing approaches to compositional entailment, including BaRDa prompting and fine-tuned approaches. Knowledge disillation from GPT-4 (RDTE Zero-Shot) improves student models by 5-21 points, at times outperforming GPT-4 itself. †/‡ denote the student models and the best performing analogous non-distilled models. Batching decompositions by their shared hypothesis does not drastically impact performance by GPT-4. Batching was not possible for BaRDa, which has one item per hypothesis.

4.6 Related Work

4.6.1 Annotating Textual Entailment Datasets

The PASCAL RTE Challenge (Dagan et al., 2005), SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018) have led to numerous NLI leaderboards. These datasets (A) are generally not tied to downstream reasoning tasks; and (B) commonly target uncontroversial items for which annotators have high agreement without needing granular instructions. The BaRDa dataset (Clark et al., 2023) throws out a large fraction of decomposition annotations because of low agreement. In contrast, RDTE specifically targets the hard-to-evaluate items that unavoidably manifest during systematic reasoning algorithms. Our use of ordinal annotation draws from JOCI (Zhang et al., 2017), a collection of common sense inferences annotated on a 5-point scale of likeliness.

4.6.2 Annotator Disagreement

Various works have faced the challenge of annotator disagreement for reasoning tasks. AmbiFC (Glockner et al., 2021), concerned with whether a piece of evidence supports a given claim, explores the common factors causing disagreements. Pavlick and Kwiatkowski (2019) found persistent patterns of disagreement amongst RTE judgments not simply due to noise; efforts such as the UNLI protocol (Chen et al., 2020)

aim to address this concern by modeling annotations on a logistic probability scale.

4.6.3 Computational Argumentation

Existing work has explored methods for computational argumentation; one subfield is argumentation mining (Palau and Moens, 2009), which aims to detect and analyze the arguments in a passage. Jin et al. (2022) collect a dataset of items exhibiting 14 types of fallacies. Stab and Gurevych (2017) hand-annotate the sufficiency of argumentative essays using RAS criteria but are not geared towards entailment.

4.6.4 Assessing Explanations

Our work builds on literature evaluating the role and quality of model-generated explanations (Wiegreffe and Marasovic, 2021; Tan, 2022). We add to this discourse by investigating how to improve the quality of decompositional entailment items that drive reasoning for complex tasks. Similar to ours is the EEV methodology (Valentino et al., 2021), under which an explanation is translated into logical form and then assessed by a trained semanticist. RDTE requires neither the translation nor the trained semanticist.

4.7 Conclusion

The trustworthy application of LLMs to complex reasoning tasks critically depends on accurate responses and justifications. To date, research in entailment tree reasoning has largely focused on measuring the former.

This work develops a protocol based on works in informal logic for the assessment of compositional entailment, the core reasoning task undergirding entailment trees. We employed this rubric-based protocol to annotate RDTE, a high-quality dataset of judged compositional entailments, along with tens of thousands of automatically scored items by GPT-4 under the same instructions in multiple domains. We found a large gap in performance between nearly all models and human performance on the RDTE dataset; however, we found that knowledge distillation provides an avenue for training more precise classifiers, which is particularly desirable in a reasoning domain for which false positives are more catastrophic than false negatives. The combination of our rubric, manual annotations, and GPT-4 derived data represents a significant advance in building trustworthy AI systems capable of not only solving complex reasoning tasks but also providing correct justifications along with their answers. IN Chapter 5, I will illustrate the effects of the RDTE project on a downstream entailment engine; RDTE-based reasoning and knowledge distillation serve as the core of the engine's ability to produce high-quality answers that are right for the right reasons.

4.8 Discussion

The RDTE dataset is a high-quality set of 1000 decompositions across two specific QA domains. As argument sufficiency is a domain-dependent notion, we had extensive discussion about what constituted validity in the two different tasks. Applying the RDTE protocol to new domains will likely also merit careful consideration of how the various facets of the task manifest differently for different types of questions.

Chapter 5

TreeWise: A Flexible, Robust Entailment Engine over Noisy Text Corpora

5.1 Motivation and Overview

The NELLIE system described in Chapter 3 and Weir et al. (2024a) is a T5-based entailment engine that shows high performance on Science QA by checking whether answer hypotheses are compositionally entailed by the WorldTree knowledge base (Xie et al., 2020). Its search algorithm suffers from the primary drawbacks that (1) it requires a clean corpus of knowledge sentences, which is not always available for a particular problem domain, and (2) its various modules all rely on different in-domain training datasets not typically available for new tasks.

My progress developing the NELLIE system in late 2022 and 2023 coincided with a paradigm shift in the broader NLP community: away from small-to-medium-sized, fine-tuned LMs such as T5 to increasingly larger LMs such as GPT-3, LLama, ChatGPT, and GPT-4 that can all leverage few-shot learning (also known as in-context learning) and instruction tuning. These models outperform the previous generation of LMs on many popular NLP reasoning tasks while requiring only a handful of in-domain training examples. This shift thus brought a more extensible and customizable series of tools to modular reasoning frameworks such as the entailment engine considered in this thesis. Functionalities already implemented in the engine such as converting an answer option into a hypothesis, proposing decompositions, and verifying entailments, all came under the purview of simple prompting mechanisms rather than requiring fine-tuning datasets (I have already illustrated the effect on the verification task in the previous chapter),

and new functionalities became not only possible but easy to rapidly prototype.[1]

In this chapter, I introduce TreeWise, an evolution of Nellie based on instruction-tuned LLMs like ChatGPT and GPT-4. TreeWise introduces a series of improvements to the engine's search algorithm that allow it to handle novel domains (1) with noisier knowledge sources like an index over Wikipedia passages common to state-of-the-art question-answering systems and (2) without module-specific training data. In this way, TreeWise can answer whether a hypothesis from a novel QA dataset is compositionally entailed by Wikipedia. I have also replaced many dataset-trained components with a series of prompting-based modules that leverage recent advances in instruction tuning for LLMs, thus reducing the reliance on domain-specific subtask training data (e.g. for entailment classification and decomposition generation) that hinders Nellie's extension to new problems.

I reimplement the Nellie system into a new engine, TreeWise to (among many improvements) leverage more powerful reasoners with stronger entailment classification abilities and the ability to reason over noisier knowledge sources such as Wikipedia. Building upon the Nellie engine introduced in Chapter 3, TreeWise now leverages in-context learning, forward chaining, branch consolidation, and most importantly, improved decompositional entailment recognition. It surpasses tree-producing approaches on established benchmarks like EntailmentBankQA, but also adapts to complex tasks such as HotpotQA that require reasoning over less structured

[1]Hence the rise of "Prompt Engineer" as a lucrative professional title (Ma and Woods, 2024).

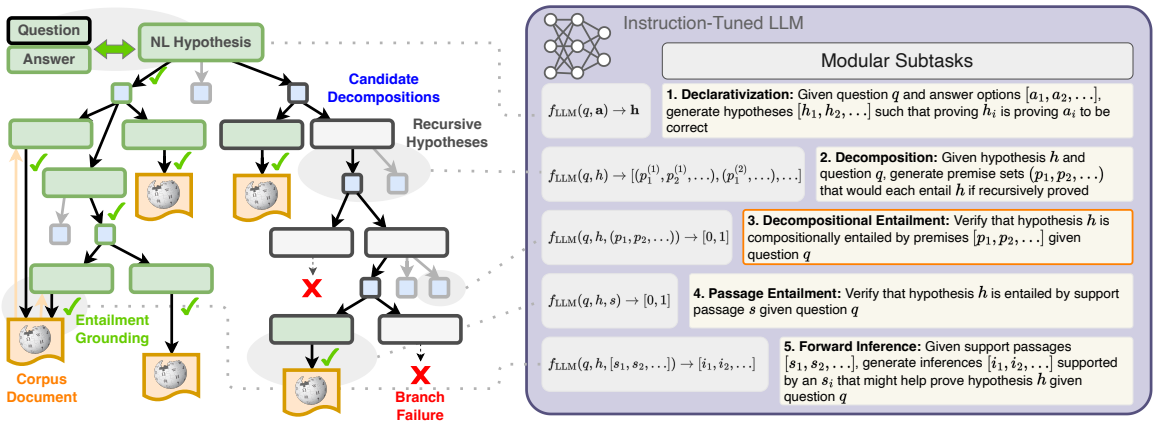


Figure 5.1: Like NELLIE, the TREEWISE engine generates many premise decompositions of a hypothesis and checks whether any candidates are valid entailments. Premises are then recursively decomposed until it finds any tree(s) fully grounded in one or more documents. Unlike NELLIE, these documents can be long, noisy passages from a corpus like Wikipedia rather than just a clean knowledge base of facts. Statements entailed by documents are generated via *forward chaining*, while the rest of the search is *backward* like in NELLIE. Many decompositions end up untraversed due to the search budget or nonentailment.

knowledge sources like Wikipedia. I also find that training an RDTE-oriented entailment classifier via knowledge distillation and employing it in the TREEWISE entailment tree reasoning engine significantly improves both accuracy and proof quality, illustrating the practical benefit of this advance for textual inference.

5.2 TREEWISE System Overview

TREEWISE builds upon the backward chaining entailment tree search framework first introduced by NELLIE (Weir et al., 2024a). The core functionality of TREEWISE is to answer the question, "is NL hypothesis H compositionally entailed by a corpus of documents C in the context of a question Q?" Illustrated in Figure 5.1, the search

attempts to ground the hypothesis via recursive decomposition into premises entailed by passages in a verifiable corpus such as Wikipedia. Implemented in Prolog, the algorithm follows a breadth-first strategy, performing the following at each recursive hypothesis H:

1. **Retrieve** a set of s support documents from a corpus like Wikipedia that are likely to contain information relevant to the hypothesis. Check if any of these documents entails H using a series of single-premise entailment classifiers. If so, H is proved. See §5.3 for implementation details.

2. **Check whether H is a paraphrase** of a previously considered hypothesis H' using SBERT (Reimers and Gurevych, 2019). If so, I associate it with the proof branches of H'.

3. **Forward generate** i inferences from the support documents via an instruction-tuned LLM, leveraging its long context to consider a large list of documents in one pass.

4. **Decompose** H into a set of d candidate decompositions using a series of prompts to the LLM. I use a diverse set of prompts to propose candidates that encourage using the i inferences.

5. **Condense the decompositions** by deduplicating semantically equivalent candidates.

6. **Filter the decompositions** using a series of decompositional entailment classifiers to weed out those that would not logically support H.

7. **Recurse on the premises** to continue the search.

This process is repeated until either a user-specified expansion budget is exhausted or a proof is found and no search branches remain that could yield a higher score. The system is designed to be amenable to a costly API; it batches together decompositions of a single hypothesis into one prompt and then asks it to score them all at once. In the following section, I will delve into more specific revisions of the search logic to implement these changes.

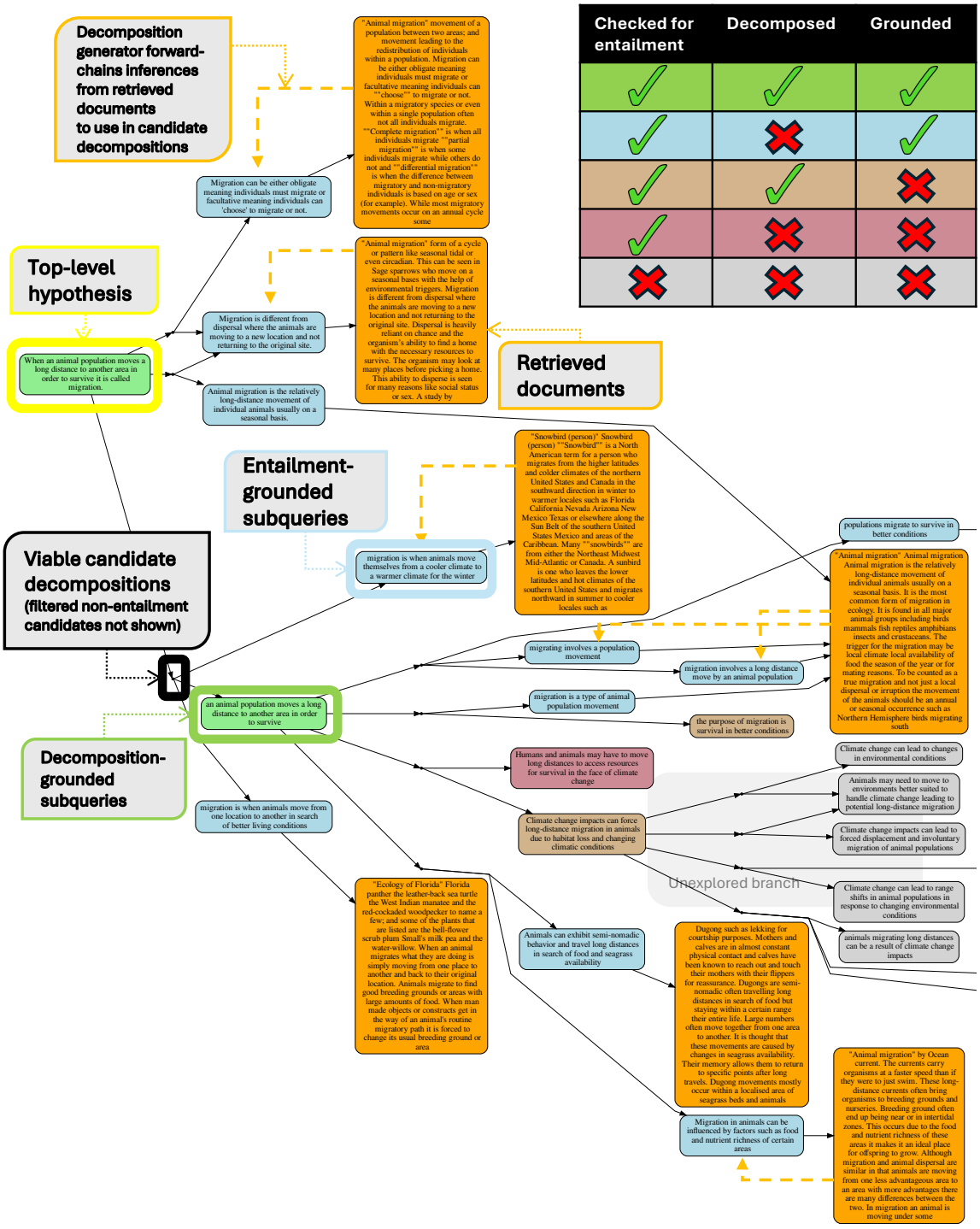


Figure 5.2: Example partial TREEWISE search trace. Green and blue nodes are grounded to Wikipedia documents in orange. The full search is limited to a budget of 60 decompositions (green and brown nodes).

5.2.1 Search Logic

I refer readers to Chapter 3 for an overview of the original NELLIE search algorithm for compositionally grounding a hypothesis in a corpus. That search generally follows a breadth-first search across candidate decompositions by following 3 Prolog rules:[2]

1. *Fact Unification*

   $\textsc{Prove}(h) \Leftarrow \textsc{Retrieve}(h^+, f^-) \wedge \textsc{Entails}(f, h)$

2. *Two Premise Rule Generation*

   $\textsc{Prove}(h) \Leftarrow \textsc{RuleGen}(h^+, f_1^-, f_2^-) \wedge \quad \textsc{Entails}([f_1, f_2], h) \wedge \textsc{Prove}(f_1) \wedge \textsc{Prove}(f_2)$

3. *Retrieved First Premise Rule Generation*

   $\textsc{Prove}(h) \Leftarrow \textsc{Retrieve}(h^+, f_1^-) \wedge \quad \textsc{RuleGen}(h^+, f_1^+, f_2^-) \wedge \textsc{Entails}([f_1, f_2], h) \wedge$

   $\textsc{Prove}(f_2)$

I make the following observations:

1. Rules 2/3 constrain the search to only binary conjunctions; allowing 3 or 4 might allow for added flexibility.

2. The LM-calling RULEGEN predicate only ever accepts 0 or 1 support facts; conditioning on multiple retrieved candidate facts might produce higher quality and/or more groundable decompositions.

[2]Following common practice in Prolog implementations, e.g., SWI-Prolog (Wielemaker et al., 2012), I assume Prolog terms are evaluated in the sequence they are read.

3. Rule 3 assumes that items returned by RETRIEVE (f_1^-) are of the same type as the premises that constitute an entailment: in NELLIE's case, simple evidential sentences like "birds can fly." If TREEWISE's corpus contains noisier knowledge, e.g., Wikipedia paragraphs, then this assumption becomes problematic.

4. Rules 2/3 assume that any unique f_1, f_2, or h is semantically distinct from other statements considered during the search. This implies that recursively calling PROVE on a new statement is never a waste of time. In practice, however, NELLIE frequently considers hypotheses that are paraphrases of each other. This risks wasting substantial search time, especially if neither the hypothesis nor its paraphrase will ever be grounded.

To address these issues, I make the following modifications, replacing rules 2 and 3 with rules 4, 2*, 3*. The latter two rules are only executed if 4 does not succeed. Bolded symbols denote string lists.

4. *Paraphrase Unification*

$$\textsc{Prove}(h) \Leftarrow \textsc{Expanded}(g^-) \wedge \quad \textsc{Paraphrase}(h^+, g^+) \wedge \textsc{Prove}(g)$$

2*. *Non-Conditioned Rule Generation*

$$\textsc{Prove}(h) \Leftarrow \textsc{RuleGen}(h^+, [], \mathbf{f}) \wedge \quad \textsc{Entails}(\mathbf{f}, h) \wedge \textsc{MapList}(\mathbf{f}^+, \textsc{Prove})$$[3]

3*. *Retrieval-Conditioned Rule Generation*

$$\textsc{Prove}(h) \Leftarrow \textsc{Retrieve}(h^+, \mathbf{s}^-) \wedge \quad \textsc{InferenceGen}(h^+, \mathbf{s}^+, \mathbf{i}^-) \wedge$$

[3] https://www.swi-prolog.org/pldoc/man?predicate=maplist/2

$$\textsc{RuleGen}(h^+, \mathbf{i}^+, \mathbf{f}^-) \ \wedge \ \textsc{Entails}(\mathbf{f}, h) \wedge \quad \textsc{MapList}(\mathbf{f}^+, \textsc{Prove})$$

The novel predicate EXPANDED is a nondeterministic function that evaluates to true iff g^- was a previous input to RULEGEN during the search. The predicate PARAPHRASE uses SBERT (Reimers and Gurevych, 2019) cosine similarity to identify paraphrases. The predicate INFERENCEGEN is a **forward chaining** inference generator that receives a list of support items (facts, passages, or otherwise) and returns a list of sentential premises likely entailed by the items that might be helpful to prove h. The new RULEGEN now accepts an arbitrary list of candidate premises ($\mathbf{i}$) and returns an arbitrary-length decomposition ($\mathbf{f}$). As a result, the premises in the decomposition might not have appeared in $\mathbf{i}$ or $\mathbf{s}$; this adds flexibility to the generator but also means that the algorithm has to call PROVE on all generated premises. If $f \in \mathbf{f}$ does appear in $\mathbf{i}$ or $\mathbf{s}$, it is likely immediately grounded via fact unification (rule 1).

5.2.2 Prompt-based Modules

With the introduction of instruction-tuned LLMs like ChatGPT and GPT-4, we find it no longer necessary to train certain NELLIE models via supervised learning on in-domain datasets. I replace the modules for query declarativization (Demszky et al., 2018), decomposition generation, and 1- and multi-premise entailment filtering with a mixture of in-context learning and zero-shot instruction prompts to a GPT model. This includes the predicates RULEGEN and ENTAILS above. All prompts can be found later in the chapter in the relevant sections.

5.2.2.1 Improved Decomposition Generators

A key improvement of TreeWise over its predecessor is the redesigned approach to generating and filtering candidate decompositions for decompositional entailment. Nellie's original decomposition generator is a T5-based model trained via supervised learning on data from a specific domain. It is difficult to (A) adapt to new domains without retraining and (B) convey to the model that decompositions serve a reasoning-related purpose. With the introduction of instruction-tuned LLMs, this becomes more straightforward. For TreeWise, I replace the T5-based generator with a diverse series of prompts for generating ad-hoc decompositions of a hypothesis in any domain:

- A **fact-conditioned** decomposition prompt (Figure 5.3) that receives a list of forward-chaining inferences derived from support documents using a separate prompt (Figure 5.5) and returns a list of candidate decompositions.
- A **follow-up generation** decomposition prompt that receives the output of the previous prompt and an instruction to revise them to better fit the given hypothesis and question
- An **in-context learning** decomposition prompt (Figure 5.4) dynamically constructed by retrieving exemplars from a fixed set of training items using BM25. I use as our training set the gold-annotated inferences from EntailmentBank (Dalvi et al., 2021).

Together these prompts populate an initial horizon of candidate decompositions to

be condensed, checked for semantic equivalence, and subsequently filtered for argument validity.

5.2.2.2 Forward Inference Generator

Similar to the approach for generating and verifying decompositions through separate algorithm modules, I detach the generating of candidate forward-chaining inferences from documents from verifying them through single document entailment and/or further decomposition. Prior to generating decompositions in Rule 3*, I use the prompt in Figure 5.5 to suggest $n = 10$ inferences from a list of $k = 30$ retrieved support documents. These inferences are *not* yet checked for entailment from the supporting documents, as filtering for directly entailed inferences at this stage risks removing useful statements that are groundable via further decomposition.

```
You are a reasoning system that searches for proofs of a hypothesis by decomposing into simpler
     premises.

For the following hypothesis and source documents, write a set of independent inferences entailed by
     one or multiple documents. The inferences should resemble world facts and should help to decompose
      the hypothesis into component reasoning steps. The inferences should NOT simply restate the
     hypothesis.

Your output should be a serialized json item, one per line, with the format {{"inference": <inference
     text>, "source": [<indices of source documents>]}} and nothing else.

QUESTION:
{question}

HYPOTHESIS:
{hypothesis}

SOURCE DOCUMENTS YOU MIGHT PULL FROM:
{documents}

{n} INFERENCES THAT MIGHT SUPPORT HYPOTHESIS:
```

Figure 5.5: Prompt used for creating forward-chaining inferences from retrieved source documents

Fact-conditioned decomposition generation prompt:

```
You are a reasoning system that searches for proofs of a hypothesis by decomposing into simpler
     premises.

Given a question and corresponding hypothesis, you give a list of {n_candidates} possible
     decompositions of the hypothesis into two or three facts, F1 and F2 (and possibly F3), such that
     proving the list of Fs would amount to proving the hypothesis through compositional entailment.
     There should be minimal "information loss" between the hypothesis and the Fs; you are looking for
     strict entailment.

Each decomposition should be some combination of core scientific principles as well as conclusions
     about the question at hand. They should not imply each other, i.e. none of them should start with
     "thus" or "therefore".

You also optionally take a list of facts that you might use in your decompositions.

Your {n_candidates} decompositions should follow different "reasoning patterns." Try to create
     decompositions that are semantically distinct and make use of different core facts or underlying
     principles.

Your output should be a serialized json item, one per line, with the format {{"fact1": <fact1>, "fact2
     ": <fact2>, "fact3" : <fact3, if necessary>}} and nothing else.

QUESTION:
{question}

HYPOTHESIS:
{hypothesis}

FACTS YOU MIGHT USE, IF THEY ARE RELEVANT:
{facts}

{n_candidates} DIFFERENT POSSIBLE DECOMPOSITIONS, ONE JSON ITEM PER LINE:

=============

(on followup)
how could we make these better? regenerate the 20 decompositions so that they are higher-fidelity and
     are better explanations for the hypothesis.
```

Figure 5.3: Prompt used for generating fact-conditioned ad-hoc decompositions. The same prompt is re-used with an additional follow-up instruction to generate better decompositions.

Exemplar-conditioned decomposition generation prompt:

```
You are a reasoning system that searches for proofs of a hypothesis by decomposing into simpler premises.

Given a question and corresponding hypothesis, you give a list of 20 possible decompositions of the hypothesis into two or three facts,
      F1 and F2 (and possibly F3), such that proving the list of Fs would amount to proving the hypothesis through compositional
      entailment. There should be minimal "information loss" between the hypothesis and the Fs; you are looking for strict entailment.

Each decomposition should be some combination of core scientific principles as well as conclusions about the question at hand. They
      should not imply each other, i.e. none of them should start with "thus" or "therefore".

You also optionally take a list of facts that you might use in your decompositions.

Your 20 decompositions should follow different "reasoning patterns." Try to create decompositions that are semantically distinct and make
       use of different core facts or underlying principles.

Your output should be a serialized json item, one per line, with the format {"fact1": <fact1>, "fact2": <fact2>, "fact3" : <fact3, if
      necessary>} and nothing else.

QUESTION:
An ecosystem is a community of organisms interacting with their physical environment. Why are decomposers an important part of ecosystems
      ? (A) They break down dead organisms to return nutrients to the soil. (B) They produce their own food for survival. (C) They play a
       role in preventing weathering and erosion. (D) They provide most of the energy to consumers.

HYPOTHESIS:
Decomposers are an important part of ecosystems because they break down dead organisms to return nutrients to the soil.

4 DIFFERENT POSSIBLE DECOMPOSITIONS, 2 OR 3 FACTS EACH, ONE JSON ITEM PER LINE:
{"fact1": "a decomposer breaks down dead organisms to return nutrients to soil", "fact2": "nutrients in soil are important for an
      ecosystem"}
{"fact1": "decomposition is when a decomposer breaks down dead organisms", "fact2": "decomposition is when a decomposer recycles /
      returns nutrients / nitrogen from dead organisms to the soil by eating those dead organisms", "fact3": "nutrients in soil are
      important for an ecosystem"}
{"fact1": "a decomposer breaks down dead organisms to return nutrients to soil", "fact2": "nutrients in soil are important to plants", "
      fact3": "plants are a part of an ecosystem"}

QUESTION:
What is the role of decomposers in a food chain? (A) They consume other organisms. (B) They break down dead organic matter. (C) They use
      the Sun's energy to make food. (D) They convert inorganic matter into organic matter.

HYPOTHESIS:
The role of decomposers in a food chain is they break down dead organic matter.

2 DIFFERENT POSSIBLE DECOMPOSITIONS, 2 OR 3 FACTS EACH, ONE JSON ITEM PER LINE:
{"fact1": "an organism is a source of organic matter", "fact2": "decomposer is a kind of role in the food chain process / in an ecosystem
      ", "fact3": "decomposition is when a decomposer breaks down dead organisms"}
{"fact1": "an organism is a source of organic matter", "fact2": "the role of decomposers in the food chain process is to break down dead
      organisms"}

QUESTION:
{question}

HYPOTHESIS:
{hypothesis}

20 DIFFERENT POSSIBLE DECOMPOSITIONS, 2 OR 3 FACTS EACH, ONE JSON ITEM PER LINE:
```

Figure 5.4: Prompt used for creating exemplar-conditioned decompositions. Exemplars are retrieved from the EntailmentBank training set using BM25 with the question and hypothesis as query.

5.2.2.3 Reasoning Filters

Our RDTE-oriented prompting strategy discussed in the main body of the paper is shown in Figure 4.7, 4.8, and 4.9. The zero-shot variant contains the same directions but no exemplars. I use a separate set of exemplars for Hotpot vs ARC. Following the RDTE protocol, these filters rate decompositions on an ordinal sufficiency scale from 1-5. We retain only the entailments scored a perfect 5/5.

In addition to this improvement at multi-premise compositional NLI, I also implement a single-premise/passage entailment rubric for use by TreeWise and when computing our tree integrity score. This prompt rates entailment on an ordinal 1-5 scale analogous to the RDTE protocol for compositional entailment; the prompt is shown in Figure 5.6.

```
QUESTION:
{question}

HYPOTHESIS:
{hypothesis}

My student was trying to prove this hypothesis as it relates to the question. He pulled up this support
      document.

In the context of the QUESTION, does the PASSAGE entail the HYPOTHESIS? In other words, could we
     reasonably infer that the HYPOTHESIS is true in the context of the QUESTION using only the
     information in the PASSAGE?

PASSAGE:
{passage}

Please only make a judgment about whether the HYPOTHESIS is entailed by the PASSAGE, and not whether it
      answers the QUESTION.
Please score it on a scale of 1 to 5:

1: Definitely not entailed-- PASSAGE has nothing to do with the HYPOTHESIS
2: Poorly entailed-- PASSAGE might be on topic but does not provide any evidence for the HYPOTHESIS
3: Moderately entailed-- PASSAGE provides some evidence to suggest the HYPOTHESIS is true, but there is
      substantial missing information or ambiguity
4: Strongly entailed-- PASSAGE provides strong evidence for the HYPOTHESIS, but there is any amount of
     missing information or ambiguity
5: Definitely entailed-- PASSAGE provides strong evidence for the HYPOTHESIS, and there is no missing
     information or ambiguity
```

Figure 5.6: Prompt used to filter and score passage-hypothesis entailment pairs.

5.3 Retrieval Index

We build retrieval indices for Wikipedia using the `pyserini` package (Lin et al., 2021). The index for Wikipedia differs between the QA datasets considered. For the below experiments on HotpotQA, we index all the data as given in the original paper. For EBQA, we index the 2021-01-20 version of Wikipedia with 100-word chunks. We use the BM25 algorithm (Robertson et al., 1995a) for first-stage retrieval and rerank using SentenceTransformer's `ms-marco-MiniLM-L-12-v2` (Reimers and Gurevych, 2019). We retrieve the top 1000 documents, rerank them, and return the

top 30 candidate support facts per hypothesis, using the concatenated question and hypothesis as our retrieval query.

5.4 TreeWise Experiments

5.4.1 Entailment Tree QA Task Definition

I pursue a version of evidence-grounded, explainable QA in which models are not only right but right for the right reasons. This is distinct from the scenario considered in Chapter 3, where only QA correctness was evaluated. Now, I consider the following task: given corpus C, multiple-choice question Q and hypotheses $h_1, \ldots, h_m$ corresponding to answer options $a_1, \ldots, a_m$, I task models to predict correct answer a_i^* and generate entailment tree T_{h_i} rooted at h_i such that (1) the T_{h_i}'s leaves are in $C \cup Q$ and (2) the tree's steps explain how each parent follows compositionally from its children. Models are evaluated on selecting a_i^* and the correctness of the reasoning in T_{h_i}.

5.4.2 Datasets

I evaluate TreeWise and a series of entailment tree-creating baselines on the two QA datasets used to construct RDTE: 340 test questions from the EntailmentBank (**EBQA**), and 419 HotpotQA questions recast as multiple choice by using GPT-4 to generate incorrect answer options.

As TreeWise can flexibly hook up to different types of knowledge sources, I evaluate it on EBQA using two different scenarios: (1) EBQA using the clean factbase WorldTree (Xie et al., 2020) as the knowledge source, and (2) EBQA using an index over English Wikipedia as the knowledge source. For HotpotQA, I only use that task's specific Wikipedia index as the knowledge source.

5.4.3 TreeWise Hyperparameters

In the TreeWise algorithm, I prompt the decomposition generators to propose 10 decompositions each, resulting in 40 candidate decompositions per hypothesis. To improve the initial search horizon, I double this number at depth 0 only. I prompt the forward-chaining prompt to produce 30 inferences entailed by retrieved documents. I use temperature=.2 sampling for all entailment filters and temperature=1 for generating decompositions.

I set the expansion budget to 80 nodes and the filter entailment filter threshold to 0.6. I define paraphrases as having SBERT cosine similarity of 0.9 or higher.

5.4.4 Metrics

I take a two-pronged approach to evaluating systems: they should produce both strong QA accuracy **and** coherent and logically sound entailment trees explaining the chosen answer. To evaluate the latter, I introduce a new model-based **tree integrity**

score. I use GPT-4 to score each of a tree's component entailment steps under the RDTE protocol. Following the intuition that an argument is only as strong as its weakest link, I take the minimum score as the tree's overall score.

5.4.5 Methods and Baselines

A core improvement of the TreeWise engine is a fine-tuned ChatGPT instance trained to perform RDTE-style sufficiency classification for decompositions. The RDTE knowledge distillation framework suggests a way forward for improving entailment engines such as TreeWise in new domains: first extract non-optimized reasoning traces from the engine, annotate them under the RDTE protocol with GPT-4 (which is itself too slow and expensive to use in the engine), train student models on the silver data, and then substitute the student as an entailment classifier in the engine. The experiment below emulates this scenario and shows that the resulting runs of TreeWise improve on both QA and on generated tree quality, highlighting the effectiveness of applying RDTE-trained distillation student models to entailment tree creation algorithms.

I compare a version of TreeWise using the RDTE-trained student ChatGPT and RoBERTa models to an identical version that uses an ICL-prompted ChatGPT and the RoBERTa model used by Nellie, which is trained on non-RDTE entailment data. I use ChatGPT as the decomposition generator and QA2D model for TreeWise, meaning it does not use GPT-4 for anything at test time. I also evaluate a pair of

greedy baselines for entailment tree creation. These baselines are designed to mimic TreeWise's behavior in a simplified manner without the systematic search:

- An **end-to-end** tree generator that retrieves one set of facts and then uses ChatGPT to decode an entailment tree in one fell swoop.
- A **stepwise** tree generator that iteratively retrieves support facts and then decodes one step of the entailment tree until the tree is fully grounded or a maximum number of steps (10) is reached.

2 depicts the pseudocode for the end-to-end tree generation baseline. 3 depicts the stepwise version. Each process is repeated 5 times, yielding 5 different candidate trees for each answer choice, then fed to the **tree integrity** scorer using the student ChatGPT to score each tree. I take the highest-scoring tree and the corresponding answer as the final output. I compare these to non-RDTE versions where the student ChatGPT is replaced by regular ChatGPT.

5.4.6 Results

Table 5.1 shows QA results for these methods. We observe that for all methods, the tree integrity score increases when using the knowledge-distilled student. For all but 1 baseline on HotpotQA, QA accuracy increases by 1 to 7 points. We observe that TreeWise achieves the highest QA and integrity scores, while the stepwise

Algorithm 2 Greedy End-to-End Entailment Tree Generator

procedure GREEDYENDTOENDPROVE(Q) $\rightarrow \leq t$ scored trees for each query in Q:
  $R = []$
  forall $q_i \in Q$ **do**
    Retrieve a set of support facts S **conditioned on** q_i**:**
    $S =$ RETRIEVESUPPORTFACTS(q_i)
    Generate m **candidate trees using ICL Prompt:**
    $T_1, \ldots, T_t =$ GENERATECANDIDATETREES(q_i, S, m, t)
    forall $T_j \in T$ **do**
      Prune disconnected nodes from the tree:
      $T_j =$ PRUNETREE(T_j)
      Check for ungrounded leaves not in S**:**
      $U =$ FINDUNGROUNDEDLEAVES(T_j, S)
      forall $l_k \in U$ **do**
        if ISENTAILEDBYINDEX(l_k)
        then $U = U \setminus \{l_k\}$
      end for
      Retain tree if all leaves are grounded or entailed:
      if $U = \emptyset$
      then R += T_j
    end for
  end for
  grade the trees using ChatGPT (student) and return best:
  return $\arg\max_{T \in R}$ SCORETREE(T)

outperforms the end-to-end generator under integrity but vice versa for two QA scenarios.

		EBQA on WorldTree		EBQA on Wikipedia		HotpotQA on Wikipedia	
Method	**Silver?**	**QA**	**Tree Score**	**QA**	**Tree Score**	**QA**	**Tree Score**
TREEWISE	Yes	**79.2**	**75.2**	**73.2**	**74.7**	**51.3**	**66.6**
	No	72.8	71.6	69.0	71.9	46.1	62.9
Stepwise Generator	Yes	66.4	69.5	56.8	68.3	47.5	58.8
	No	63.7	68.1	51.5	64.7	49.4	58.6
End-to-End Generator	Yes	68.7	66.4	57.6	66.4	48.5	59.1
	No	68.7	66.1	55.1	64.0	43.0	57.7

Table 5.1: QA and tree integrity score for tree-generating approaches with vs without silver knowledge distillation.

Algorithm 3 Stepwise Entailment Tree Generator

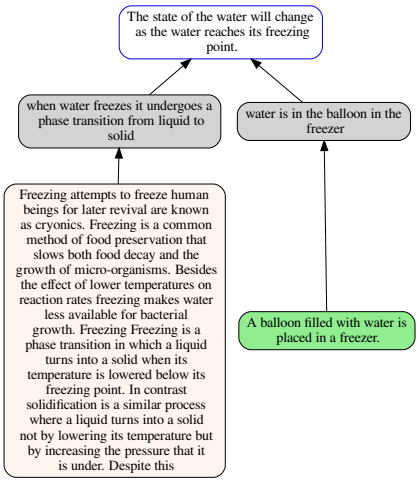
procedure StepwiseProve(Q) $\rightarrow$ t scored trees for each query in Q:
  $R = []$
  forall $t \in \{1, 2, \ldots, t\}$ **do**
    forall $q_i \in Q$ **do**
      Initialize search frontier and decompositions:
      $F = \{q_i\}$, $D = []$, $N = 0$
      while $F \neq \emptyset$ **and** $N < 10$ **do**
        $N = N + 1$
        Retrieve and flatten support facts for sentences in F:
        $S =$ Set(Flatten(RetrieveSupportFacts(f) for f in F))
        Generate one line of tree decomposition using ICL prompt:
        $d_N =$ GenOneStep(q_i, S, D)
        Append line to decompositions:
        $D = D + [d_N]$
        Create tree from decompositions:
        $T_N =$ CreateTree(D)
        Prune disconnected nodes from the tree:
        $T_N =$ PruneTree(T_N)
        Check for ungrounded leaves:
        $U =$ FindUngroundedLeaves(T_N, S)
        forall $l_i \in U$ **do**
          if IsEntailedByIndex(l_i, f)
          then $D = D + [\text{“}l_i \Leftarrow f\text{”}]$, $U = U \setminus \{l_i\}$
        end for
        Set F to be the remaining ungrounded leaves:
        $F = U$
      end while
    end for
  Retain T_N
  R += T_N
  end for
  grade the trees using ChatGPT (student) and return best:
  return $\arg\max_{T \in R}$ ScoreTree(T)

5.4.7 Example TreeWise Outputs

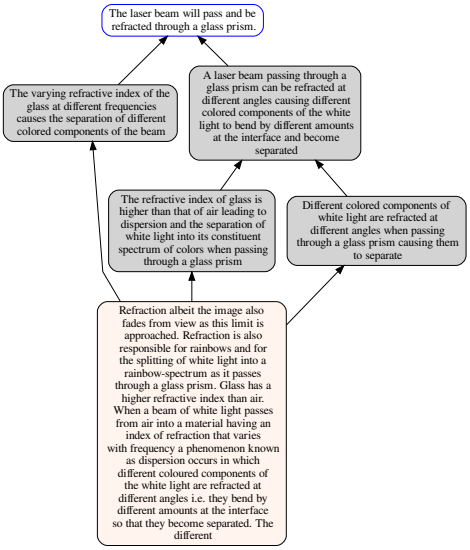
Figure 5.7 and Figure 5.8 depict example outputs by the TreeWise tree search algorithm when hooked up to Wikipedia as the knowledge source for answering ARC

and HotpotQA questions.

A balloon filled with water is placed in a freezer. Which property of the water will change as the water reaches its freezing point? (A) color, (B) mass, **(C) state**, (D) weight



A laser beam is aimed at four different objects. Through which of these objects will the laser beam pass and be refracted? (A) a black cloth, (B) a piece of aluminum, (C) a sheet of paper, **(D) a glass prism**



Which of these is not an instinctive behavior? (A) a bird building a nest, (B) a turtle burying its eggs, (C) a bear hibernating in winter, (D) a horse pulling a plow

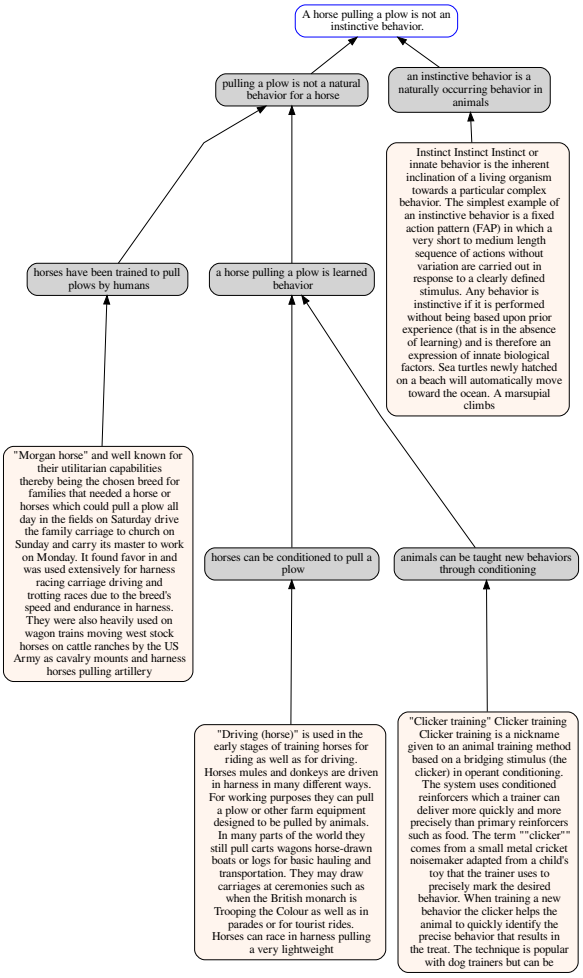
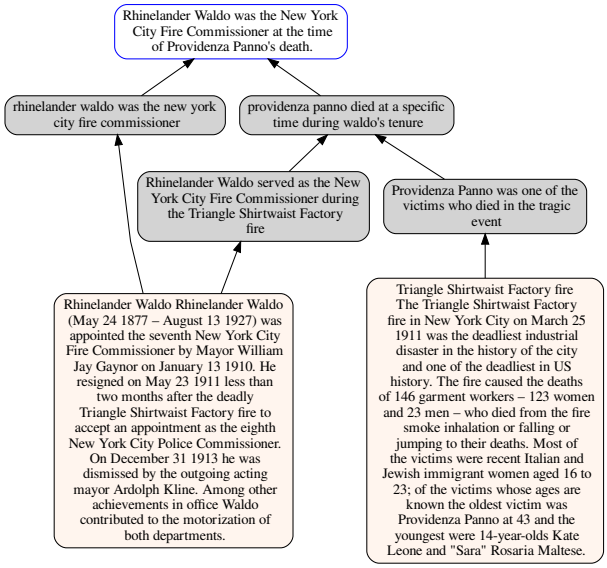


Figure 5.7: Example multiple-choice questions from ARC with NELLIE's answer and corresponding proof grounded in Wikipedia.

Who was the New York City Fire Commissioner at the time of Providenza Panno's death? (A) Nicholas Scoppetta, (B) John J. Scannell, (C) William K. King, (D) Albert M. Arroyo, (E) Charles A. La Guardia, (F) Rhinelander Waldo, (G) James E. Langdon



Which Air Force member was behind enemy lines for 11 1/2 days and had the largest,longest and most complex rescue mission?

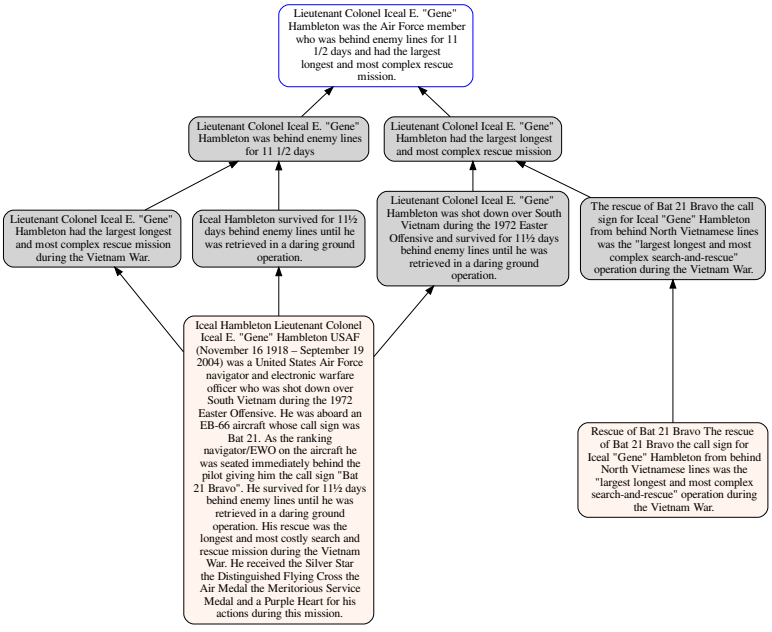


Figure 5.8: Example multiple-choice questions from HotpotQA with NELLIE's answer and corresponding proof grounded in Wikipedia.

5.4.8 Sensitivity to Search Configuration

A recurring challenge I have faced when applying NELLIE and TREEWISE to question-answering domains is the trade-off between reasoning precision and proof recall. A substantial motivating factor in pursuing the RDTE project described in Chapter 4 was the observation that the T5-based NELLIE model commonly produced (1) proofs of incorrect hypotheses and (2) proofs of correct hypotheses that have faulty reasoning. We showed in Section 4.5 that it is possible to train higher-precision decomposition filters using a careful, granular prompting protocol. This reduces the extent of false positives in the recursive search algorithm that leads to both (1) and (2). However, this can damage the engine's performance if it fails to populate the search horizon with enough *good* decompositions.

In developing TREEWISE for question answering, my initial experimental runs with the higher-precision reasoning filter knocked the precision-recall pendulum so far in the direction of precision that the search commonly failed to identify any proofs within the provided search budget of 30.

In November 2023, I made a series of key changes to the TREEWISE configuration reflected in the final QA performance displayed in Table 5.1:

- Substantially increased the search budget from 30 to 80, and **implemented a per-question budget schedule** that reaches 80 expansions only if no proof is found for any answer option. Given 4 answer options, we start with an expansion budget of 20 per option. We try to prove every option. If we do not prove a

single option, we increase the budget by 10 and start again (API and retrieval calls are cached, so only the additional expansions do not cost any time). We continue this loop until either any answer is proved or we hit a budget of 80. Most correct hypotheses are proved within the first 20-30 expansions. This means the loop will break quickly most of the time, but for questions that require longer searches, this scheduling mechanism leads to still finding proof.

- Implemented **filter precision backoff**: If after 80 expansions per option TreeWise still does not find a proof, I lower the threshold by which the entailment filter accepts candidates and rerun the search. Rather than accepting only 5/5-rated entailments, I let it accept 4/5 ones as well. This is purely a QA-oriented solution, as I don't consider a tree with GPT-rated 4/5 inferences to be a perfectly grounded proof. This is meant only to get TreeWise to produce *some* answer prediction when it otherwise would not.

- **Expanded the branching factor**: TreeWise was previously limited to exploring just 2 filter-accepted expansions per recursive hypothesis, causing the search to spend more time trying to prove subqueries at deeper depths. As proving higher-up hypotheses are more likely to lead to a proof, I set the recursive expansion budget to only 4.

- **Added the forward-chaining module** described Section 5.2.2.2, which facilitated **raising the number of facts considered during decomposition**

generation. Previously, the decomposition generator accepted 15 documents that could potentially be used for decomposition. With the forward chaining mechanism as an intermediate processing step, I now retrieve **30** documents and provide them to the forward chainer, which produces 10 inference statements to pass along to the decomposition generator.

The result of these changes is shown in Figure 5.9.

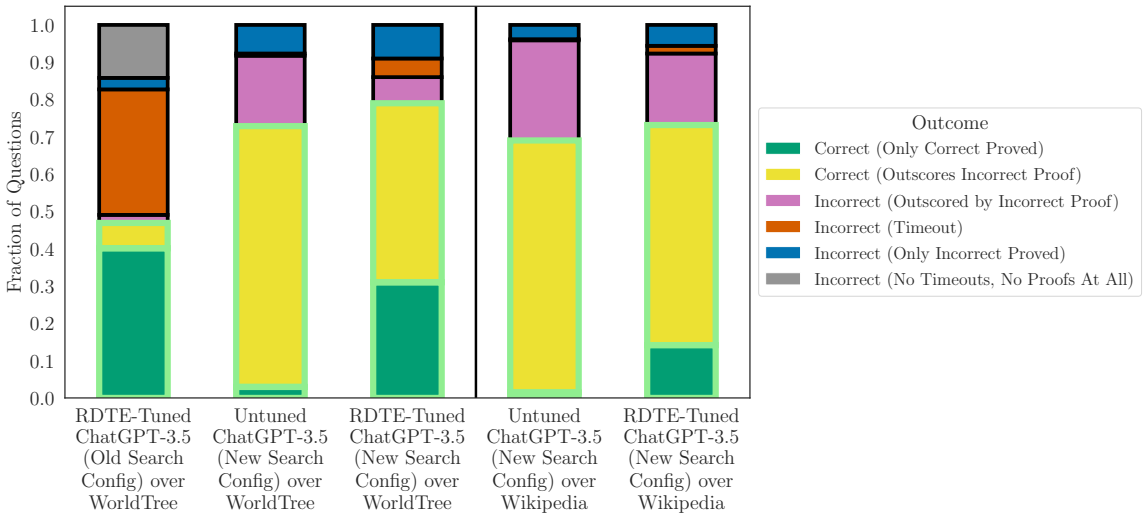


Figure 5.9: Effect of the changes in search configuration and decompositional entailment model on TreeWise's performance on EBQA using WorldTree and Wikipedia as the underlying knowledge corpus. Runs are broken down by end scenario, with the bar portions outlined in light green corresponding to questions that the engine answers correctly. The rate at which only the correct answer is proved increases when using the RDTE-tuned ChatGPT instead of regular ChatGPT (column 2 vs 3 and 4 vs 5). However, this might not correspond to higher QA accuracy, as under the previous search configuration column 1) the rate at which both the correct and at least one incorrect are proved is lower, and the search for a proof of the correct hypothesis times out very frequently. With the additions of an expanded search budget, budget and filter precision scheduling, and forward chaining from larger retrieval sets, we achieve higher QA accuracy.

5.5 Discussion

In this chapter, I presented an evolution of the NELLIE entailment engine called TREEWISE. The new engine utilizes a next generation of instruction-tuned, prompt-based LLMs in addition to the lessons and techniques learned in the RDTE annotation project in order to flexibly reason over noisy sources of knowledge such as Wikipedia. In doing so, this is the first system to my knowledge that allows users to answer the question: "does Wikipedia entail my hypothesis?" This opens up the opportunity to apply entailment engines to problem domains for which we don't have a clean knowledge store of facts such as WorldTree. It lets us apply TREEWISE to new domains such as in Chapter 6, where I illustrate its application to medical question answering by hooking it up to a corpus medical textbooks.

My experiments evaluated TREEWISE and a series of entailment-tree-generating baselines on two axes of success: (1) answering multiple-choice questions correctly, and (2) answering them using correct reasoning traces. To evaluate the latter, I introduced an automatic tree evaluation metric based on eliciting judgments from strong LLMs under the RDTE protocol. I demonstrated that TREEWISE not only outperforms the baseline systems, but also that improving the system's entailment recognition capabilities via RDTE knowledge distillation yields state-of-the-art results by TREEWISE for evidence-grounded entailment tree generation.

Chapter 6

Distilling Microtheories from Language Models

In this chapter, I will turn from considering the *search* component of entailment engines to the underlying *knowledge* component against which query hypotheses are grounded. A regular need in classical AI was to manually assemble collections of facts to facilitate and ground reasoning. One such effort was to create different "microtheories" to contain the core, generalizable kernels of facts and axioms needed by a hypothesis prover to reason in a particular context. Microtheories were appealing for symbolic reasoning because they provided clear, concise, and readily available distillations of the core principles underlying a topic. They served as pieces of "computational clockwork" reflecting how (part of) the world behaves, with statements that could work together to answer many questions.

In the era of LLMs, we have models that have implicitly captured a lot of knowledge about how the world behaves, though it is stored implicitly. In this chapter, I consider (1) whether we can automatically construct explicit microtheories from large language models and (2) consider the ways in which a microtheory framework can benefit the neuro-symbolic entailment-based hypothesis provers described in Chapter 3 and 5. I first introduce a question-driven extraction technique that populates a knowledge store similar to a popular, hand-constructed resource (WorldTree). I then demonstrate how these automatically generated microtheories benefit the performance of modern entailment-based reasoners at question answering.

6.1 Motivation

This chapter pursues a modern take on the notion of a *microtheory* (Mt): a "bundle of assertions" that provide a mechanism for focusing on relevant information during problem solving (Blair et al., 1992). Mts were introduced in the foundational ontological engineering project, Cyc (Lenat et al., 1990). They are meant to capture a core, generalizable, self-consistent representation of a problem domain to facilitate reasoning. A domain might be "physics," "how money operates," or even "brushing your teeth". In the Cyc formulation, Mts comprised a list of first-order logic facts and rules that bootstrapped Cyc's symbolic hypothesis-proving algorithm. In modern AI, researchers generally take for granted that large language models (LMs) have parametrically memorized the kinds of assertions one would expect to store in a Mt (Petroni et al., 2019), and can explicitly or implicitly or use them for reasoning. Accordingly, "such frameworks [as Mts] play little or no role in current Big Data-centric approaches to AI" (Marcus and Davis, 2019), and efforts such as Cyc to handwrite, or scrape (Schubert, 2002; Mitchell et al., 2018), vast commonsense knowledge ontologies for which Mts serve as organizing principles have been abandoned in favor of LLMs. The so-called "knowledge acquisition bottleneck" has been seemingly solved by data-hungry models trained to memorize text on the web.

However, we argue that microtheories can provide important benefits to modern LM-based reasoners. LLM-generated reasoning traces are usually not grounded in externally verified knowledge and may include hallucinations (Ji et al., 2022), which

hinders their widespread adoption in areas such as medicine and law where precise reasoning is paramount and faulty evidence is impermissible (Lohr, 2023). What would be desirable is to have some sort of explicit structure or index of knowledge serving as an interface between LLMs' implicitly stored knowledge and the users that rely upon them. Having a discrete and accessible representation of the knowledge an LLM uses to make decisions allows practitioners the ability to query, vet, and isolate incorrect knowledge kernels offline and provides a mechanism for reasoning robustly over the representation, the ability to add to and improve it.

In addition to verifiability, we see three other kinds of benefits that a discrete microtheory might provide to a neural LLM-based reasoner as it did symbolic ones, illustrated in the context of entailment-based reasoners in Figure 6.1:

- **Efficiency**: symbolic microtheories were designed to provide computational efficiency, speeding up symbolic theorem provers by giving them the most generalizable facts and rules for a problem domain. We envision similar gains for LLM-based reasoning, whereby providing relevant knowledge-rich domain statements can amortize some of the reasoning required for new problems.

- **Grounding**: filling in potential domain-relevant knowledge gaps of LMs and serving to ground reasoning without LMs having to first memorize and the regurgitate all necessary facts

- **Seeding** more accurate reasoning algorithms using true and useful statements

in complex domains, thereby producing better quality traces and higher task accuracy.

To illustrate these effects, we consider the recent class of entailment tree-based interpretable reasoning algorithms (Dalvi et al., 2021; Tafjord et al., 2022; Weir et al., 2024a). Similar to symbolic algorithms grounding a hypothesis to a symbolic "theory" of facts and rules, these "entailment engines" ground a *natural language* (NL) hypothesis to NL 'theory' of verified NL knowledge via forward- and backward-chaining search. These works commonly consider the WorldTree knowledge store, a hand-annotated corpus of 12K science facts that help explain the answers to grade-school science questions and a resource that we argue constitutes a version of a modern, NL microtheory for grade-school science. Using WorldTree as grounding knowledge yields strong performance by entailment tree reasoners, who use it as the 'leaves' of their proof-like trees explaining how they arrive at a hypothesis via entailment. As WorldTree is manually crafted and is limited to a single topic domain, we ask whether such a resource could be created *automatically* for new domains.

We introduce a question-driven methodology for automatically extracting a WorldTree-like Mt for any given problem domain. This involves prompting LMs to enumerate facts used to solve training hypotheses, then identify the core, abstractable pieces of knowledge constantly reused for many questions in a dataset, for which we use an LM-based entailment engine. This process inevitably faces challenges of noise due to repetition and semantic equivalence, for which we introduce a series of techniques

for distilling a pool of assertions down to the most powerful and reusable. We show via human analysis that the resulting Mts contain predominantly generalizable and useful facts in multiple problem domains (grade school science, medical question answering).

We then examine the effect of providing an automatically extracted Mt to an entailment engine at test time when answering new questions. In the recently proposed task of Entailment Tree Question Answering (described in Chapter 5), in which a model must both answer a question correctly and also provide a valid tree showing how the answer hypothesis follows from knowledge in an external NL text source such as Wikipedia, we show that providing an mT to the engine as additional knowledge for grounding and forward chaining improves the engine's performance to a similar extent to using the WorldTree corpus, and can do so for new domains not covered by WorldTree such as Medical QA.

6.2 Background

6.2.1 Symbolic Microtheories

In symbolic logic, a deductive *theory* is a collection of assertions in a formal language divided into *facts* and *rules* from which a deductive reasoner can make inferences. While these inference engines can be powerful in applied settings, curating the right theories for any given problem is challenging, which has spurred decades of AI research designing and implementing *ontologies* for representing symbolic knowledge. The Cyc

project was an ambitious effort to encode all the implicit common-sense knowledge that everybody knows into a single knowledge base. Cyc encoded two million axioms, organized into 6000 internally consistent *microtheories*.

Given a hypothesis to prove, the Cyc deductive inference reasoner would identify the relevant Mt, thereby isolating the most relevant knowledge in its vast, common-sense knowledge base, and then perform a combination of backward chaining (identifying rules that would explain the hypothesis and recursively considering the rules' conditions) and forward chaining (taking statements from the Mt and applying rules to make new inferences that are relevant to the problem). The designers of Cyc, in part due to the challenges of scaling such a symbolic search, emphasized *concision*: "In general, a microtheory should be small enough to have a reasonable set of assumptions that most of the assertions in the context depend on" (Blair et al., 1992).

Microtheories thus exhibited the three benefits illustrated in Figure 6.1: They improved **efficiency** by making an automated theorem proving search tractable and effective with respect to a problem domain, provided **completeness** by providing the core facts needed to ground a symbolic hypothesis, and improved **reasoning** by generally allowing the model to answer more queries correctly.

Despite its generally unmatched size, scope, and lofty ambitions, the Cyc project is generally viewed as a failure, having yielded a level of reasoning performance that was unable to match those of smaller, simpler systems in competitions such as Project Halo (Friedland et al., 2004). Part of the reason for this is due to what Sowa calls the

challenge of "knowledge soup" (Sowa, 2006): no static ontology can be perfect for all problems and circumstances because the knowledge in a human's head is too noisy and inconsistent to be written down accurately enough for a deductive reasoner to exactly emulate it. Microtheories, as Sowa (1993) describes, were the most promising avenue towards addressing this challenge, creating small, tightly organized theories "floating in the soup" making the knowledge base more modular and extendable by multiple authors to new problems by assembling multiple microtheories. Yet, he noted the numerous open research challenges related to organizing microtheories: "How are the microtheories related to one another, how are they combined or modified to form new theories, and how does the system choose which theory to use to answer any particular question?"

6.2.2 Language Models as Sources of Microtheory-like Knowledge

The widespread adoption and development of large language models (LLMs) by AI communities have opened new opportunities to answer these questions. LLMs show to memorize and capably regurgitate all sorts of the world and common sense knowledge upon careful prompting and exhibit strong performance on knowledge-intensive benchmarks such as MMLU, which suggests a strong handle on the application of world knowledge in many technical domains. LLMs provide a dynamic knowledge

store of common sense (Ilievski et al., 2021; Weir et al., 2020) and world knowledge: a version of Sowa's knowledge soup, "a large reservoir of loosely organized encyclopedic knowledge" stored within an LM's neural parameters that can be queried via recent advances in prompting. The most performant LMs such as GPT-4 (OpenAI, 2023) have been adopted as capable replacements for human annotators for datasets and evaluations (Zheng et al., 2023) and approach human performance on challenging benchmarks such as MMLU (Hendrycks et al., 2021) that test for esoteric world knowledge in various disciples.

The flexibility of LMs presents us with an intriguing possibility: while, according to Sowa, no static ontology exists that can handle all domains, LMs might let us *dynamically* construct a collection of assertions that provides the same benefits as a symbolic microtheory, retrieving relevant knowledge from the LM's continuous representation and facilitating forward- and backward-chaining as needed by a deductive reasoner. Given a new NLP task dataset, an LLM-based system could "choose" its microtheory by extracting it as needed from its implicitly stored knowledge.

What might a microtheory actually look like for LLM-based reasoners? Notably, LLMs have the capacity to reason in natural language as opposed to symbolic formalism; this bypasses many of the challenges faced by the community in defining a language suitable for representing and reasoning over the knowledge necessary to solve hard problems. Accordingly, we will explore microtheories in the form of NL instead of symbolic facts and rules. To illustrate this, we examine the case of entailment engines,

which prove hypotheses against theories of natural language.

6.2.3 Entailment Engines

Much as in symbolic logic, an entailment engine performs a soft version of theorem proving that grounds a query hypothesis H to a theory $\mathcal{T}$. Both elements are stated in NL: H is a sentence (commonly a declarative answer to a question Q), and $\mathcal{T} = [t_1, \ldots t_n]$ is a corpus of text passages of that can vary in length from sentences to entire documents. Unlike symbolic logic, the engine does not rely upon a domain-specific set of inference rules written down in an ontology. Instead, it relies on the soft logic of textual entailment determined by data-driven neural models. The entailment engine's job is to determine whether $\mathcal{T}$ *entails* H via a compositional entailment tree T.

Formally, we define the function $\textsc{engine}(H, \mathcal{T}; Q) \rightarrow [T_1, T_2, \ldots]$, where each T_i

- is rooted at H
- has leaves $L_i = [l_1, \ldots, l_l] \subseteq \mathcal{T}$
- is composed of composition steps taking the form $[p_1, \ldots p_j] \models_Q h_k;\ j \in \{1, \ldots, |T| - 1\}$, where $\models_Q$ refers to natural language entailment given the context of question Q.

Each inference step has a corresponding model-provided confidence score. Existing works, including those discussed in Chapter 3 and 5, assign each T_i the lowest score amongst its inferences.

Chapter 4 provides an operational definition for $\models_Q$ using tenets from informal logic: $P \models_Q h$ iff the premises in P are a sufficient argument to a target human audience that h should be accepted as true in the context of Q. Transitively, we can surmise that the leaves L_i provide an *argumentative basis* for accepting H. If the engine generates multiple trees with different leafsets, we can say that $L_1, L_2, \ldots$ constitute alternative bases that sufficiently support H.

6.2.3.1 Extending Microtheories for Neural Entailment

The neural engine described above is a departure from existing systems capable of symbolic textual entailment reasoning (e.g. Nutcracker (Bos, 2014) or EPILOG (Schubert et al., 2010)), for which proof search relies upon handwritten general axioms specifying allowable inferences, e.g. (from Bos and Markert (2005)),

$$\forall e \forall x \forall y (\text{event}(e) \wedge \text{in}(e, y) \Rightarrow \text{in}(x, y)) \tag{6.1}$$

which states that if an event is located in y, then so is the agent of that event. Neural engines like NELLIE or TREEWISE instead rely upon neural models to validate entailment and meta-rules that specify the structure, but not specific instances, of Horn clause inference rules, e.g.

$$\forall x \forall y \forall z \forall Q (\text{fact}(x) \wedge \text{prove}(y) \wedge ([x, y] \models_Q z) \Rightarrow \text{prove}(z)) \tag{6.2}$$

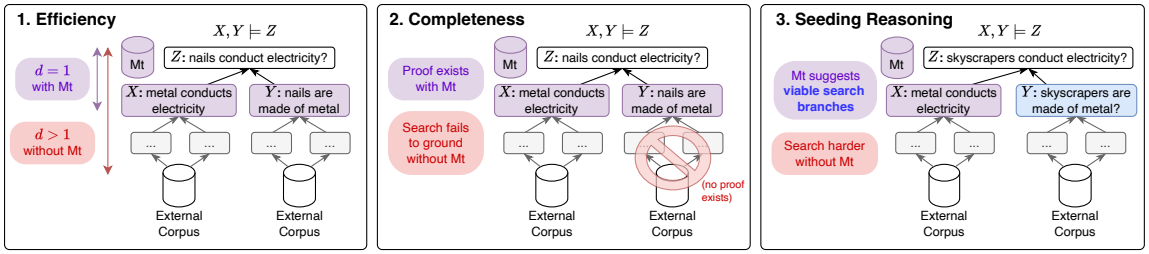


Figure 6.1: Illustration of the desired benefits that adding a microtheory as a source of knowledge should provide an entailment engine.

An instantiation of this might be

- $x =$ "if an event is located in y, then so is the agent of that event"
- $y =$ "Penny is walking in Baltimore"
- $z =$ "Penny is in Baltimore"

This accomplishes the same role as the symbolic axiom using just an axiom-like NL sentence as x and a neural entailment model for $\models_Q$.

Moreover, while symbolic engines require symbolic facts e.g., isCountry(USA), the neural entailment engines handle such knowledge via the same representation as they do rules: NL text, e.g. the sentence "The USA is a country", or alternatively, a longer document entailing that the USA is a country such as its Wikipedia page.

We thus take a different approach to defining a Microtheory for the neural entailment engine: rather than *separately* pursue facts and general rules, we can pursue them simultaneously through the shared medium of rich, inference-supporting natural language statements.

6.2.3.2 How Microtheories can help Entailment Engines

While this shift away from symbolic rules alleviates the challenge of hand-engineering carefully crafted axioms, it introduces a new set of search efficiency-related problems. State-of-the-art symbolic provers, e.g. Vampire (Riazanov and Voronkov, 2002b), can search fairly quickly through a purely symbolic search space defined by an ontology and find a solution within a reasonable time budget if there is one. From Bos (2014): "the bottleneck for the logical approach to RTE isn't the current state of automated semantic analysis or theorem proving, but the lack of supporting background knowledge."[1] The capacity to handle knowledge stated in NL rather than symbolic formalism reduces the burden of acquiring background knowledge.

Yet, the switch to NL explodes the search space. While symbolic search is delineated by which symbols can unify, entailment engines rely upon a version of "weak unification" between NL statement symbols (Weber et al., 2019), under which any two symbols can unify with some unification score determined by a model (in our case, an entailment model for $\models_Q$). Because any two symbols might unify, the model must pick which subset of its vocabulary are the most viable proofs. For example, in rule 2, x can unify with any fact in a very large NL corpus, while y can unify with not only the NL corpus but also with any other statement generated by the backward-chaining algorithm to be recursively proved. Repeatedly calling neural models to generate candidate ys and

[1] As noted in Chapter 2, this position is at odds with some of the findings of the Lighthill Report, which identified combinatorial explosion as an inevitable result of scaling up symbolic search to hard problems. For the purposes of symbolic entailment search, however, it seems to have held for Bos.

verify $\models_Q$ is much slower than hard unification, which is near instantaneous. This makes the search budget a paramount concern for the functioning of the entailment engine algorithm. One way an Mt might benefit an engine is by reducing the burden of the search space, making statements more viable for proving new hypotheses that don't have to be recursively proved.

6.2.3.3 WorldTree: a Microtheory for Entailment Engines

For a given topic, we seek the core pieces of knowledge, stated in natural language, that best contribute (partially, if not completely) to the argumentative bases that entail hypotheses in that domain. We take inspiration from WorldTree (Xie et al., 2020), a resource constructed for a similar purpose in ScienceQA and used substantially as a knowledge source in Chapter 3 and 5. WorldTree is a semi-structured, manually-annotated knowledge base of over 9K facts organized into 66 inference-supporting tables that "provide a substantial portion of the knowledge required to answer non-spatial, non-mathematical elementary science questions." (Jansen et al., 2018).[2] Subsets of facts from WorldTree serve as argumentative bases for ARC answers. Various approaches to question answering, both entailment-based, such as TreeWise, and otherwise (Valentino et al., 2022) have found success using it as a knowledge source via information retrieval. The inference-supporting kinds of facts in WorldTree, therefore,

[2]We note that WorldTree actively contains two types of facts: so-called "central" facts ("freezing means matter changes from a liquid to a solid by decreasing heat energy"), as well as more idiosyncratic grounding and lexical facts ("to slow down means to decrease speed") that are less general. The automatic construction approaches described in §6.3 are catered towards the former category, which is more similar to the style of concise, generalizable Mts targeted by Cyc.

provide us an archetype to follow when constructing microtheories automatically.

6.3 Extracting a Microtheory

As described below, we translate the microtheory construction process into the context of entailment engine-based reasoning to the NL assertions that an engine can most commonly use to correctly answer questions for a given task dataset. We propose to use an LLM, in conjunction with an entailment engine, to dynamically instantiate this set of assertions. For a given dataset $[Q_1, \ldots Q_d]$, we extract from an LM, via prompting, sets of facts that it would use to support the correct answer to each question (§6.3.1). As this leads to a pool of highly redundant and potentially ungeneralizable statements, we apply a series of optimizations to distill down the core kernels of knowledge more concisely by identifying entailment relations (§6.3.2) and optimizing for fact usage by the engine (§6.3.3).

6.3.1 Raw Fact Pool Extraction

We first prompt an LM to regurgitate facts on a question-by-question basis to populate an initial factpool $\mathcal{F}$:

$$\mathcal{F} = \bigcup_{1}^{d} \text{Decontextualize}(\text{TheoryCoT}(Q_i)) \tag{6.3}$$

We refer to the prompt as a "TheoryCoT" one because it cues the LM to perform a version of 'chain-of-thought' reasoning (Wei et al., 2022) catered towards extracting facts. It generates a list of WorldTree-like statements for Q_i (a question-specific 'theory'), then an entailment tree that uses a subset of facts in the list, and finally, an answer to the question. The prompt (shown in Figure 6.2) is a few-shot learning prompt using exemplars dynamically retrieved[3] from EntailmentBank. Each exemplar's fact list comprises the gold WorldTree facts from the unstructured explanation graph, and its entailment tree is the gold entailment tree that uses a subset of the facts. Thus, while Q_i might not be in the same ScienceQA domain as the exemplars, we cue the LM to generate *WorldTree-like* facts that should serve similar roles as argumentative bases in entailment trees. We discard the model-generated entailment tree and take the generated fact list as candidates to be added to $\mathcal{F}$.

[3]We use BM25 (Robertson et al., 1995b) for retrieval using the question text as the query as done for declarativization and ICL-based decomposition generation in Chapter 5.

```
You are an expert theory generation system that lists facts about a problem area that will help solve a reasoning question. Given a
       question, you generate QUERY statements for each of the possible answers to the question. The queries should be complete statements
        that exactly match the answer choices as closely as possible.

Then, you generate a theory comprising simple FACTS about the world. These should be statements that you believe to be logical and true
       about the world that will help you prove the queries. A FACT is either a generic statement about the world ("birds can fly") or a
       specific statement about the context that helps support the proof of the query ("Tweety is a bird").

For a QUERY, you generate at least 6 statements.

Once you generate the theory, you show how one of the QUERYs logically follows from the context. The selected QUERY corresponds to the
       answer choice that you believe is correct. You can only choose one QUERY as correct.

To show your reasoning, you generate a PROOF of your chosen QUERY hypothesis. Your PROOF is an "entailment tree" that composes the FACTS
       of your theory in order to show how the query is compositionally entailed by them. Each step in the tree is a composition of two or
        more premises into a new conclusion, which should be used as a premise in future steps. The last step compositionally proves the
       hypothesis. When you reuse the new conclusion in a later step, you refer to it by its id. Each step is separated by a newline.

###
QUESTION: Which of the following actions is most likely part of a test to find the hardness of a mineral sample? (A) heating the sample
       on a hot plate (B) scratching the sample with a nail (C) hitting the sample with a hammer (D) shining a bright light on the sample

QUERIES:
(QUERY A) Heating the sample on a hot plate is most likely part of a test to find the hardness of a mineral sample.
(QUERY B) Scratching the sample with a nail is most likely part of a test to find the hardness of a mineral sample.
(QUERY C) Hitting the sample with a hammer is most likely part of a test to find the hardness of a mineral sample.
(QUERY D) Shining a bright light on the sample is most likely part of a test to find the hardness of a mineral sample.

THEORY:
(FACT 1) a mineral is a kind of material
(FACT 2) hardness is a measure of a mineral 's ability to resist scratching
(FACT 3) an example of breaking something down is scratching something
(FACT 4) a material that is soft can be broken down by a material that is hard
(FACT 5) testing often requires measuring
(FACT 6) a nail is a kind of object
(FACT 7) hardness is a property of a material / an object and includes ordered values of malleable / rigid
(FACT 8) hardness can be used to identify minerals
(FACT 9) a mineral is a kind of solid / natural material
(FACT 10) nails are often made of iron
(FACT 11) scraping an object may cause small particles to break off of that object
(FACT 12) if a mineral can be scratched by a fingernail then that mineral is soft
(FACT 13) hardness is a kind of physical property
(FACT 14) hardness is an intensive property
(FACT 15) soft is a kind of hardness
(FACT 16) experimentation requires measuring
(FACT 17) an event is a kind of action
(FACT 18) if a material scratches easily then that material has low hardness
(FACT 19) a mineral is a kind of object
(FACT 20) mineral nutrient is a kind of nutrient
(FACT 21) objects are made of materials / substances / matter
(FACT 22) rock is made of minerals
(FACT 23) a process is made of a series of actions
(FACT 24) if an object is made of a material then that object has the properties of that material
(FACT 25) testing / identifying the streak of a mineral requires the powdered form of that mineral

PROOF of Scratching the sample with a nail is most likely part of a test to find the hardness of a mineral sample.:
(STEP 1) a material that is soft can be broken down by a material that is hard & an example of breaking something down is scratching
       something -> int1: a material that is soft can be scratched by a material that is hard
(STEP 2) int1 & a mineral is a kind of material -> int2: if one mineral can scratch another mineral then that other mineral is softer
       than that one mineral
(STEP 3) hardness is a measure of a mineral 's ability to resist scratching & int2 -> hypothesis: Scratching the sample with a nail is
       most likely part of a test to find the hardness of a mineral sample.

ANSWER: B

###
QUESTION: {question}

QUERIES:
{queries}

THEORY:
```

Figure 6.2: TheoryCoT in-context learning prompt for extracting WorldTree-like facts from an LLM for a given question.

```
###
QUESTION: A 47-year-old woman with chronic epigastric pain comes to the physician because of a 1-month history of intermittent, loose,
      foul-smelling stools. She has also had a 6-kg (13-lb) weight loss. She has consumed 9-10 alcoholic beverages daily for the past 25
      years. Seven years ago, she traveled to Mexico on vacation; she has not been outside the large metropolitan area in which she
      resides since then. She appears malnourished. The stool is pale and loose; fecal fat content is elevated. An immunoglobulin A serum
       anti-tissue transglutaminase antibody assay is negative. Further evaluation is most likely to show which of the following? (A)
      Trophozoites on stool microscopy, (B) Pancreatic calcifications, (C) Villous atrophy of duodenal mucosa, (D) Positive lactulose
      breath test

QUERIES:
(QUERY A) Further evaluation is most likely to show trophozoites on stool microscopy.
(QUERY B) Further evaluation is most likely to show pancreatic calcifications.
(QUERY C) Further evaluation is most likely to show villous atrophy of duodenal mucosa.
(QUERY D) Further evaluation is most likely to show a positive lactulose breath test.

THEORY:
(FACT 1) Chronic epigastric pain can be a symptom of various gastrointestinal disorders.
(FACT 2) Intermittent, loose, foul-smelling stools can be a sign of malabsorption.
(FACT 3) Significant weight loss can be a sign of malnutrition or malabsorption.
(FACT 4) Long-term, heavy alcohol consumption can lead to pancreatitis and malabsorption.
(FACT 5) Travel to certain regions can expose individuals to parasites that cause gastrointestinal issues.
(FACT 6) A negative immunoglobulin A serum anti-tissue transglutaminase antibody assay rules out celiac disease.
(FACT 7) Malabsorption can lead to elevated fecal fat content.
(FACT 8) Pale and loose stools can be a sign of malabsorption.
(FACT 9) Pancreatic calcifications can be a sign of chronic pancreatitis.
(FACT 10) Villous atrophy of duodenal mucosa can be a sign of celiac disease or other gastrointestinal disorders.
(FACT 11) A positive lactulose breath test can indicate small intestinal bacterial overgrowth (SIBO).

PROOF of Further evaluation is most likely to show pancreatic calcifications.:
(STEP 1) Long-term, heavy alcohol consumption can lead to pancreatitis and malabsorption & Significant weight loss can be a sign of
      malnutrition or malabsorption -> int1: The patient's long-term, heavy alcohol consumption may have led to pancreatitis and
      malabsorption, causing significant weight loss.
(STEP 2) int1 & Intermittent, loose, foul-smelling stools can be a sign of malabsorption -> int2: The patient's intermittent, loose, foul-
      smelling stools are likely due to malabsorption caused by pancreatitis.
(STEP 3) int2 & Pancreatic calcifications can be a sign of chronic pancreatitis -> hypothesis: Further evaluation is most likely to show
      pancreatic calcifications.
```

Figure 6.3: Example TheoryCoT output for a MedQA question. The generated proof steps are thrown out and the generic facts are added to the fact pool $\mathcal{F}$. Context-specific facts are discarded.

We observe that the LLM frequently generates context-specific sentences in the process of grounding certain hypotheses, e.g., about specific entities in the question (e.g. "Penny is in Baltimore")[4] Since we wouldn't want to add this sort of statement to our Microtheory, we create a filtering mechanism (Decontextualize() in (3)) that prompts the LLM to classify each fact as generic or context-specific and keeps only the former.[5]

[4]This phenomenon also occurs in EntailmentBank trees, where some leaves are entailed by the question text.

[5]As with any LM-based method, this process is imperfect and permits the occasional context-specific statement to be added to $\mathcal{F}$. We observe that it is often filtered by the optimization techniques below and rarely impacts downstream reasoning.

6.3.2 SBERT and Entailment-based Condensation

As fact lists are extracted independently for each question, one would expect substantial repetition within the unified pool. The same information might be stated slightly differently, and some facts might subsume one or multiple others. We thus use a pair of neural techniques to remove semantic duplicates and facts that are entailed by others.

6.3.2.0.1 Soft Deduplication

We first identify pairs of paraphrases using an SBERT bi-encoder model trained to maximize the embedding cosine similarity based on semantic textual similarity (STS).[6] We perform a linear pass of $\mathcal{F}$ and remove every fact f_i for which $\exists f_j \in \mathcal{F}, j > i, \cos(\text{SBERT}(f_i), \text{SBERT}(f_j)) > t$ for some threshold t.[7]

6.3.2.0.2 Entailment Condensation

We then perform a second pass through $\mathcal{F}$ to find pairs f_i, f_j s.t. $f_i \models f_j$, and remove f_j. We use a neural cross-encoder trained for entailment classification.[8] As it would be computationally infeasible to check all $|\mathcal{F}|^2$ pairs of facts, we leverage the SBERT bi-encoder once more to whittle down to the most promising candidates. We check for entailment between any two facts f_i, f_j iff $\cos(\text{SBERT}(f_i), \text{SBERT}(f_j)) >$

[6] `all-mpnet-base-v2`
[7] We use $t = .9$.
[8] `sileod/deberta-v3-large-tasksource-nli`

$u, u < t$.[9]

The resulting condensed factpool $\mathcal{C} \subseteq \mathcal{F}$ should contain fewer semantically equivalent statements.

6.3.3 Engine Usage-based Optimization

Even after removing duplicates and entailed facts, we observe a substantial amount of facts that are distinct but serve the same *functional* role. For example,

(A) An object's acceleration can be calculated by dividing force by mass

(B) The force acting on an object equals mass times acceleration

(A) and (B) serve the same role in many physics questions but imply slightly different things.

While this might be acceptable when populating a large, noisy knowledge store with infinite storage, we instead seek a more concise representation without functional redundancies. Towards this goal, we propose a series of approaches to identify some n most effective statements to retain in the Mt. We will refer to the resulting sets as n-Mts.[10] Under this budgetary constraint, we would not want to retain both (A) and (B) in the n-Mt if keeping both meant discarding some (C) that is important to answer other questions. The three approaches described below are illustrated in Figure 6.4.

[9] We use $u = 0.3$.

[10] n is a user-specified hyperparameter; below, we will observe that n represents a trade-off between concision and performance, and there is no singular way to discern an ideal n value.

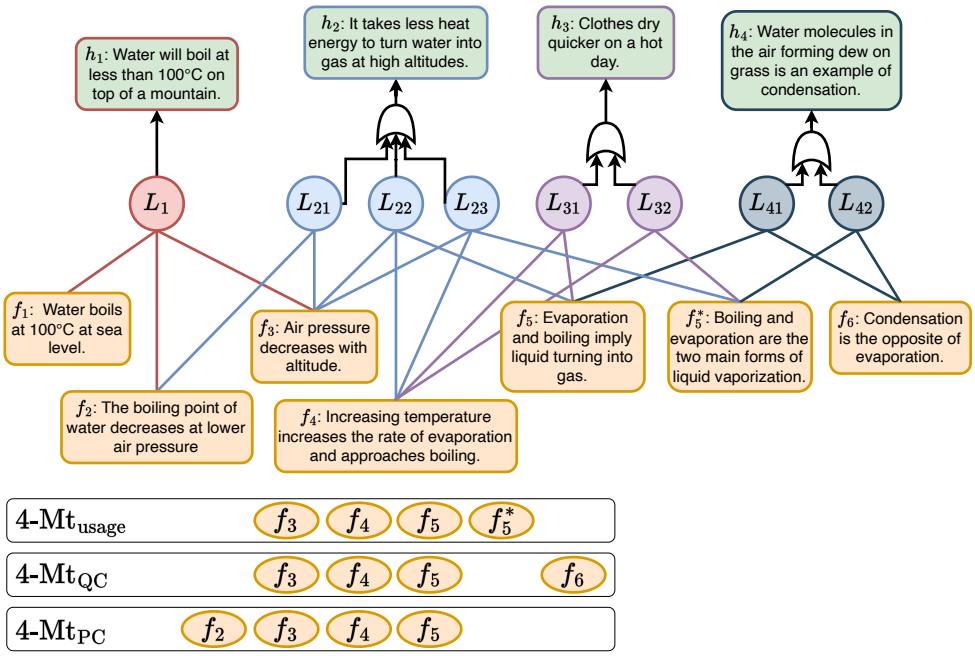


Figure 6.4: Comparison of three different optimization techniques for extracting n-Mts from a larger fact pool $\mathcal{C}$. The *usage* approach prioritizes facts based on the number of questions for which they are used. This risks keeping facts that serve the same role in explaining the same hypotheses (e.g., f_5 and f_5^* are both kept, even though only one is needed to explain h_2, h_3, and h_4). The *question coverage* (QC) approach maximizes the number of hypotheses for which the n-Mt contains *all* of a supporting argument's leaves (here the three hypotheses h_2, h_3, and h_4). This risks failing to cover some hypotheses at all (e.g. h_1). *Partial coverage* (PC) maximizes the total *fraction* of the argument covered for all questions (preferring the most covered argument for each h, if there is more than one). In this example, coverage $= 0.66{+}1{+}1{+}.5 = 3.16$ (out of a possible 4).

6.3.3.1 Top-n Most Used Facts

We seek the set of facts that best "covers" the argumentative bases for a given dataset of hypotheses. One intuitive way to choose these facts is to feed them to an entailment engine and see which ones are used most frequently to explain hypotheses in the dataset. We will refer to this approach as the *usage* n-Mt. Using training

questions $Q_1 \ldots Q_d$ and hypotheses $h_1 \ldots h_d$:

$$n\text{-Mt}_{\text{usage}} = \underset{\mathcal{M} \subseteq \mathcal{C}, |\mathcal{M}|=n}{\arg\max} \sum_{f_i \in \mathcal{M}} \sum_{j=1}^{d} \delta(f_i \in \textsc{leaves}(\textsc{engine}(h_j, \mathcal{C}; Q_j))) \tag{6.4}$$

Where LEAVES refers to the leaf set of all facts used at least once by the ENGINE in an entailment tree for h_j. However, this objective does not address the functional redundancy issue for fact pairs such as (A) and (B). If (A) and (B) serve the same role in entailing a given hypothesis, then the engine will likely produce a tree that contains (A) as a leaf, and a similar tree that replaces (A) with (B). Both facts will be in the leaf set for a similar number of questions, so if one is in the top-n most used facts, the other will likely also be.

6.3.3.1.1 Maximum Question Coverage Linear Program

We construct an objective function that maximizes the number of train hypotheses. Given each question Q_j is associated with bases $\mathcal{L}_j = L_{1j}, L_{2j}, \ldots$, we will attempt to fully cover at least one L_{kj} for as many questions as possible using only n facts.[11]

$$n\text{-Mt}_{\text{QC}} = \underset{\mathcal{M} \subseteq \mathcal{C}, |\mathcal{M}|=n}{\arg\max} \sum_{j=1}^{d} \delta(\exists L_{kj} \in \textsc{LeafSets}(\textsc{engine}(h_j, \mathcal{C}; Q_j)); L_{kj} \subseteq \mathcal{M}) \tag{6.5}$$

We implement a linear program to perform this optimization.

[11] This is a version of the maximum coverage problem in Computer Science; we have a universe of facts $\mathcal{C}$, many subsets of the universe (the Ls), and will attempt to select a n-subset of the universe $\mathcal{M}$ that contains Ls corresponding to am maximum number of questions.

6.3.3.1.2 n-MT_{QC} ILP Implementation

We set the objective function to achieve the following goals, ordered by precedence:

- Maximize the coverage of questions by at least one proof tree.
- Maximize the number of proof trees covered by the selected facts.
- Minimize the number of facts included in the new theory.

We define the objective as

$$\min\left(\sum_{i=1}^{|\mathcal{C}|} x_i - \sum_{j=1}^{d}\left(\omega z_i + \sum_{k=1}^{|\mathcal{L}_i|} y_{kj}\right)\right) \tag{6.6}$$

Where x_i is a binary variable indicating whether fact i is included in the Mt, y_{kj} is a binary variable indicating whether basis L_{kj} for question j is covered by the selected facts, and z_j is a binary variable indicator whether question q_j has at least one basis fully covered. We set ω to be sufficiently high to dwarf the other terms, using them only for breaking ties.

We impose the following constraints:

- At most n facts are kept.

$$\sum_{i=1}^{|\mathcal{C}|} x_i \leq n \tag{6.7}$$

- Each L_{kj} is "covered" iff all its facts are kept.

$$\forall L_{kj}, \quad \sum_{i=1}^{|L_{kj}|} x_i \geq y_j \cdot |L_{kj}| \tag{6.8}$$

- Each question is "covered" if at least one of its bases is "covered"

$$\forall \mathcal{L}_j, \quad \sum_{j=1}^{|\mathcal{L}_j|} y_j \geq z_k \tag{6.9}$$

6.3.3.2 Maximum Partial Coverage Linear Program

The program optimizing (5) leaves question coverage as a binary decision variable: for each Q_i, either an entire argument basis is kept, or Q_i is not covered. We consider a variant that loosens this constraint, allowing for *partial* coverage of questions.[12] We construct an objective function that maximizes the per-question partial basis coverage. Given each question Q_i associated with bases $L_{1i}, L_{2i}, \ldots$, we try to keep facts to best cover some L^*_{ki}.

$$n\text{-Mt}_{\text{PC}} = \underset{\mathcal{M}\subseteq\mathcal{C},|\mathcal{M}|=n}{\arg\max} \sum_{j=1}^{d} \underset{L_{kj}\in\textsc{LeafSets}(\textsc{engine}(h_j,\mathcal{C};Q_j))}{\arg\max} \frac{|\mathcal{M}\cap L_{kj}|}{|L_{kj}|} \tag{6.10}$$

[12]For 3 questions, it is perhaps more desirable to cover 70%, 70%, and 70% of a basis for each than to cover 100%, 100%, and 0%.

6.3.3.2.1 n-MT_{PC} ILP Implementation

We set the objective function to achieve the following goals, ordered by precedence:

- Maximize the partial coverage of each question by the selected facts.
- Minimize the number of facts included in the new theory.

We define the objective as

$$\min\left(\sum_{i=1}^{|\mathcal{C}|} x_i - \sum_{j=1}^{d} \omega p_j\right) \tag{6.11}$$

Where x_i is a binary variable indicating whether fact i is included in the Mt, p_j is a continuous variable indicating the maximal partial coverage of question j by the selected facts, and y_{kj} is a continuous variable indicating the coverage of basis L_{kj} for question j by the selected facts. z_{kj} is a binary indicator of whether a given L_{kj} is the highest coverage of question j attained by the solution.

We impose the following constraints:

- At most n facts are kept using Equation 6.7.
- Each L_{kj} is "covered" by the proportion of its facts that are kept.

$$\forall L_{kj}, \quad y_{kj} = \frac{1}{|L_{kj}|} \sum_{i \in L_{kj}} x_i \tag{6.12}$$

- Two constraints that enforce p_j equals the highest partial coverage for question j using the Big M method: The maximal partial coverage of each question is at

least as large as the coverage of each of its bases.

$$\forall \mathcal{L}_j \forall k, \quad p_j \geq y_{kj} - M(1 - z_{kj}) \tag{6.13}$$

and the maximal partial coverage of each question does not exceed the coverage of each of its bases.

$$\forall \mathcal{L}_j \forall k, \quad p_j \leq y_{kj} + M z_{kj} \tag{6.14}$$

- Exactly one z_{kj} is active per question.

$$\forall \mathcal{L}_j, \quad \sum_{k=1}^{|\mathcal{L}_j|} z_{kj} = 1 \tag{6.15}$$

6.3.3.3 Retaining Inferences

When collecting the bases L_{ij}, we want the entailment engine to best identify which facts and sets of facts are most commonly reused across questions. Since the engine has the potential to miss certain combinations, we add a mechanism to encourage it to reuse inferences made during earlier training questions $Q_1 \ldots Q_{i-1}$ when finding bases for Q_i. After each call to ENGINE$(h_i, \mathcal{C}; Q_i))$, we take all intermediate sub-hypotheses grounded by the engine to facts in $\mathcal{C}$ in the process of proving h_i, some $\mathcal{S}_i$ and add them to the fact pool for all future questions $j > i$. Whenever the engine uses any subhypothesis $s \in S_i$, we replace it with all the ways the engine proved s using bases in $\mathcal{C}$. For example, if the engine returns basis $[f_1, s]$ and s was itself previously grounded

by the sets $[f_2, f_3], [f_2, f_4]$, we replace $[f_1, s]$ with $[f_1, f_2, f_3]$ and $[f_1, f_2, f_4]$.[13]

6.4 Entailment Engine Experiments

I evaluate the quality of our constructed microtheories over two domains: **ARC**, comprising grade-school science questions, and **MedQA**, comprising medical questions from the USMLE medical school exams.

We target the ability of methods to extract sets of generalizable knowledge from training questions and then apply them readily to test questions. This requires train/test splits that are about the same topics. Due to computational efficiency issues, it is impractical to run the entailment engine over thousands of training questions. Thus, for our initial exploration of engine-supporting microtheories, I create mini-splits of ARC and MedQA that contain topically similar questions. I construct a clustering algorithm shown in Algorithm 4 that gathers questions targeting similar knowledge by iteratively prompting an LLM to identify core knowledge sentences for each question, cluster the knowledge sentences using SBERT, then prompt the LLM to "caption" the sentence clusters with topic labels. For the embedding clustering algorithm I use the SentenceTransformer-based community detection algorithm that clusters sentences greedily according to some minimum cosine similarity threshold, which I set to 0.6.

For ARC, I construct a question split containing 9 topic clusters including "Genetics

[13]As this can result in a combinatorial explosion, we limit the number of fact sets "unfolded" in this way to a total of 1000 per question.

Algorithm 4 Two-Step Clustering Algorithm for Question Grouping by Core Topic

procedure TwoStepClustering(Q) → Hierarchically clustered questions with topic and hypertopic labels:
  $core\_facts = []$
  forall $q_i \in Q$ **do**
    $core\_fact =$ LLM.ExtractCoreFact(q_i)
    $core\_facts$ += $core\_fact$
  end for
  $clusters =$ SBERT.Cluster($core\_facts$)
  $topic\_labels = []$
  forall $c_j \in clusters$ **do**
    $topic\_label =$ LLM.GenerateTopicLabel(c_j)
    $topic\_labels$ += $topic\_label$
  end for
  $topic\_clusters =$ SBERT.Cluster($topic\_labels$)
  $hypertopic\_labels = []$
  forall $tc_k \in topic\_clusters$ **do**
    $hypertopic\_label =$ LLM.GenerateHypertopicLabel(tc_k)
    $hypertopic\_labels$ += $hypertopic\_label$
  end for
  Assign $topic\_labels$ **and** $hypertopic\_labels$ **to corresponding questions in** Q
  return Q with hierarchical topic and hypertopic labels

Mini-Split	ARC	MedQA
# Questions	854/127/249	641/94/186
# Subtopics	9	4
$\lvert\mathcal{F}\rvert$	20,561	11,082
$\lvert\mathcal{C}\rvert$	8,722	8,183
External Corpus	Wikipedia	Wiki + MedQA Textbooks

Table 6.1: Dataset and fact pool statistics for the two considered domains.

and Evolution", "Force, Mass, Acceleration and Gravity", and "Plant Growth and Reproduction.". For MedQA, I construct a split containing 4 clusters including "The Effects of Smoking on Health" and "Kidney Function and Disorders."

6.4.0.1 Evaluated Methods

I construct n-Mts for n=100, 500, 1000 using the *usage*, *question coverage* and *partial coverage* approaches described in §6.3.

6.4.1 Entailment Engine Configuration

I use TreeWise (Weir et al., 2024b) as our entailment engine. For ARC, I use GPT-4 (`gpt4-0613`) as the underlying LM to extract the Mts from training questions, and ChatGPT (`gpt-3.5-turbo-1106`) as the LM for test questions. I use the fine-tuned ChatGPT instance from Weir et al. (2024b) to verify entailment steps. For MedQA, I use Mixtral-8x22B-Instruct-v0.1 (Jiang et al., 2024) as the underlying LM for both extraction and test-time inference. To collect training proofs, I explore up to depth 2 for ARC hypotheses (which are easier to prove) and up to 40 expansions for MedQA.[14]

6.4.2 Extraction Results

Table 6.1 displays various statistics about the number of questions and size of the resulting fact pools created during extraction. I set a time limit of 1 hour for each ILP run. The MedQA raw factpool $\mathcal{F}$ is only half the size of the ARC one, despite having

[14]MedQA hypotheses require greater depth proofs on average than ARC. I found that even with 40 expansions, TreeWise often timed out without finding a proof. To partially address this, I reconfigured the TreeWise engine to no longer generate all decompositions at a particular depth in parallel; this is often wasteful of the budget since for a given decomposition $f_1, f_2 \models_Q h$, there is no point in decomposing f_2 if f_1 can't be proved. There are around 10 premises on average within the first round of up to 6 (a hyperparameter) decompositions of the top-level hypothesis; by turning off parallelism, I can reallocate the 40-expansion budget to lower-level hypotheses and improve grounding.

75% the number of questions– this reflects the fact that more of the LLM-extracted facts about MedQA questions are contextual ones that pertain to the specific patient case described in each question. These are filtered out of $\mathcal{F}$ by our contextual fact identification prompt. Figure 6.5 shows histograms for how frequently each fact in $\mathcal{C}$ was used in the 650-850 questions.

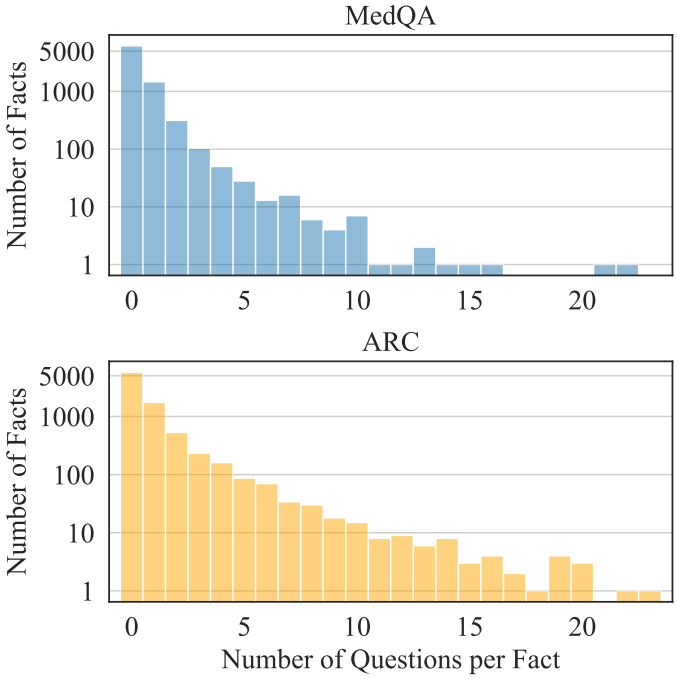


Figure 6.5: Histogram of per-fact training question usage rate for the 8000 facts in $\mathcal{C}$ and 650-850 training questions.

In both datasets, a substantial fraction of facts (5000/8000) are never used in a proof.[15] A couple dozen facts in ARC are used in more than 10 questions, which is the case for less than 10 facts in MedQA, suggesting that there are fewer singular generalizable core facts in the MedQA fact pool that provide support for many

[15]This implies that the optimization methods described in Section 6.3.3 select around 3000 facts that were used at least once in a training proof.

questions. This is unsurprising, as USMLE exams are much more complex than grade school science and require memorization of much more information that can be more esoteric. This highlights a challenge faced when attempting to apply the notion of a microtheory to a domain such as medicine that requires high levels of rote memorization and complex applied knowledge. Whether a concise microtheory will be helpful in a domain correlates at least partially with the extent to which there exists a small set of generalizable and frequently relevant facts for that domain at all. I did not consider an open domain, entity-centric domain such as Natural Questions (Kwiatkowski et al., 2019) or HotpotQA (Yang et al., 2018) for this reason; each question targets different specific world knowledge contained in Wikipedia.

Figure 6.6 and 6.7 list the top-25 most frequent facts used for training questions, as well as a sample of 10 from the long tail of facts used for only one question. I also show which of these facts are retained in the partial coverage 100-Mt$_{PC}$. The top ARC facts are very similar to the kind that appear in WorldTree; many of the facts in MedQA are about chronic obstructive pulmonary disease (COPD), reflecting that the Smoking topic is the largest in the training set and that COPD is a common medical condition associated with smoking. Multiple COPD facts are not retained in the microtheory.

Fact	#Qs	Mt_{PC}
any feature which is determined by genes that can be transmitted from parent to offspring during reproduction is considered an inherited trait	23	Yes
the earth turns on its axis from west to east once every 24 hours	22	Yes
sedimentary rocks are most often formed by the compaction and cementation of sediment, which is material transported and deposited by wind, water, or ice	20	Yes
the tilt of earth's axis is the reason for the changing amount of sunlight that each hemisphere receives, which in turn is the driver for seasons	20	Yes
genes are the instructions for physical traits that pass from parents to offspring	20	Yes
Though Earth's revolution around the Sun does affect the duration of daylight and darkness, the basic existence of the two phenomena is due to Earth's rotation on its axis	19	Yes
inheriting means receiving genetic information and traits from a parent or parents	19	No
Physical traits of an organism, such as fur color, eye color, and size, can be genetically passed from parents to offspring	19	Yes
The rotation of the Earth regulates the daily cycle of day and night, not the seasonal cycle	19	Yes
the rotation of the Earth on its axis is the movement of the Earth around a central point, once every 24 hours	18	No
Earth's rotation is the spinning of the Earth around its imaginary axis that passes through the North and South Poles	17	Yes
The Earth's rotation period is constant, providing a predictable cycle of day and night	17	No
the tilt of earth 's axis changes the angle of sunlight and thus seasonal temperatures	16	No
The rotation of Earth causes day and night and affects climate, but does not directly affect soil formation	16	No
parents pass on traits or characteristics, such as eye color and blood type, to their children through their genes	16	Yes
the force of gravity between two objects increases as the mass of the objects increases and decreases as the distance between the objects increases	16	Yes
children inherit their physical traits from both of their biological parents	15	Yes
inherited behaviors are genetically passed on from parents to offspring	15	No
the sun, moon, and stars appear to rise in the east and set in the west because of Earth's rotation	15	Yes
the force of gravity on an object is determined by its mass and its distance from the Earth's center of mass	14	Yes
the earth is tilted at an angle of about 23.5 degrees, this tilt—combined with factors such as Earth's spin and orbit—leads to variations in the duration of sunlight received on various parts of Earth's surface	14	No
The Sun's light illuminates half of the Earth at a time, creating our experience of day and night as the Earth rotates	14	Yes
As Earth rotates around its axis, different stars appear at different positions in the sky	14	Yes
Weathering and erosion are natural processes that significantly alter the shape, size, and texture of rocks	14	No
Processes needed in formation of a sedimentary rock are weathering, erosion, deposition, compaction, cementation	14	Yes
Whales utilize ocean currents and temperature gradients to guide their migration routes	1	No
Mutations can be caused by various factors such as exposure to radiation, certain chemicals, or errors during DNA replication	1	No
a tree requires sunlight to grow	1	No
lava beds are igneous rock, formed from cooled lava	1	No
wilting is a survival mechanism, limiting water loss by reducing the surface area exposed to the sun	1	No
Mendel's experiments with pea plants were focused on heredity	1	No
a flood plain is an area near a river where the land tends to flood often	1	No
Fertility in offspring is a sign of successful interbreeding within a species	1	No
The effectiveness of erosion varies with the amount and force of the eroding agent (wind, water, etc.)	1	No
glaciers often leave behind distinctive landforms and features that were created by their movement and melting	1	No

Figure 6.6: Top-25 most and 10 least used (at least once) facts in the ARC microtheory.

Fact	#Qs	Mt_{PC}
Smoking is a significant risk factor for many diseases, including chronic obstructive pulmonary disease (COPD) and lung cancer	22	Yes
Chronic obstructive pulmonary disease (COPD) is a common respiratory condition that can be caused by long-term smoking and can lead to respiratory failure and death	21	Yes
A 68-year-old man who has smoked 1 pack of cigarettes daily for 40 years is at high risk for lung cancer and chronic obstructive pulmonary disease (COPD)	16	No
Smoking is the leading cause of COPD	15	No
Decreased urine output, increased creatinine, and decreased creatinine clearance are signs of kidney injury	14	No
The patient's symptoms and laboratory findings are consistent with nephrotic syndrome, which is a condition characterized by high levels of protein in the urine and low levels of serum albumin	13	No
The patient has a history of smoking, which can cause damage to blood vessels and other organs	13	No
COPD is a progressive lung disease that makes it hard to breathe. It is caused by damage to the lungs from smoking or other factors	12	No
The longer the duration and the more packs per day a person smokes, the greater the risk of lung cancer	11	No
The patient has a history of smoking and drinking alcohol, which increases the risk of lung infections and abscesses	10	No
A urea nitrogen concentration of 42 mg/dL and a creatinine concentration of 2.3 mg/dL are both elevated and indicate kidney damage	10	Yes
The patient's BUN and creatinine levels are slightly elevated	10	No
A 50 pack-year smoking history is a significant risk factor for lung cancer	10	No
The patient has a history of smoking, which can cause lung conditions that lead to a chronic cough	10	No
The man's elevated creatinine level is another indicator of kidney damage	10	No
The disorder described in the question is caused by a mutation in an autosomal gene and is not expressed in all individuals who inherit the mutated gene, suggesting it is autosomal recessive	10	Yes
Smoking two packs of cigarettes daily for 40 years is a risk factor for lung cancer and heart disease	9	No
Symptoms of UTIs include suprapubic pain, urinary frequency, and a positive leukocyte esterase test	9	Yes
Bloody urine, also known as hematuria, can be a sign of various medical conditions, including urinary tract infections, kidney stones, bladder cancer, or kidney disease	9	Yes
Urea nitrogen and creatinine are waste products that are filtered out of the blood by the kidneys. High levels of these waste products can indicate kidney problems	9	No
The patient has a history of alcoholism, which can lead to liver damage and hepatic encephalopathy	8	No
Severe hypertension can cause damage to the kidneys, leading to high urea nitrogen and creatinine levels and proteinuria	8	No
Chronic obstructive pulmonary disease (COPD) is characterized by a persistent airflow limitation that is usually progressive	8	Yes
Autosomal recessive disorders are caused by mutations in genes on non-sex chromosomes and require two copies of the mutated gene to manifest	8	No
The presence of blood in urine with dysmorphic features and RBC casts suggests a glomerular disease	8	Yes
The Gram stain showing gram-negative diplococci suggests the presence of Neisseria meningitidis, a type of bacteria that can cause meningitis	1	No
Togavirus, Paramyxovirus, and Picornavirus can cause myocarditis	1	No
A positive quellung reaction indicates the presence of Streptococcus pneumoniae or Neisseria meningitidis	1	No
Prominent nucleoli, coarse chromatin, and multiple nuclei can be signs of cancer	1	No
Widely spaced nipples and prominent heels and convex, rounded soles are physical features that are not typically associated with any specific genetic disorder	1	No
Fat necrosis is characterized by the breakdown of fat tissue	1	No
An FVC of 78Turner syndrome is also associated with an increased risk of mitral valve prolapse, which is a condition in which the mitral valve does not close properly	1	No
Turner syndrome can cause ovarian dysgenesis, which is the failure of the ovaries to develop properly	1	No
Prader-Willi syndrome patients often have a small head size	1	No

Figure 6.7: Top-25 most and 10 least used (at least once) facts in the MedQA microtheory.

6.4.2.1 Qualitative Microtheory Examples

Figure 6.8 depicts a subgraph of a much larger network of entailments connecting statements in $\mathcal{C}$ to 18 of the training hypotheses. It shows which of these statements were retained by the partial coverage Mt for their support of multiple related hypotheses. The subgraph illustrates 4 cliques: one for hypotheses about the gravitational force between objects, and 3 about the force, acceleration, and mass of objects in motion. The core statements retained in the Mt are mainly versions of Newton's second law of physics, $F = m \cdot a$ and Newton's Law of universal gravitation. While a singular fact describing the law might theoretically serve the same purpose in proving all these hypotheses, I observe that the entailment engine prefers to use separate facts representing different interpretations of the law (e.g., how to calculate net force vs how to calculate the force required to move an object). This may also result from imposing a depth-2 limit on the training proof search. It would perhaps require more "legwork" by the entailment engine to construct a chain of more than 2 entailments from a single statement, e.g., just 'force equals mass times acceleration,' to each hypothesis.

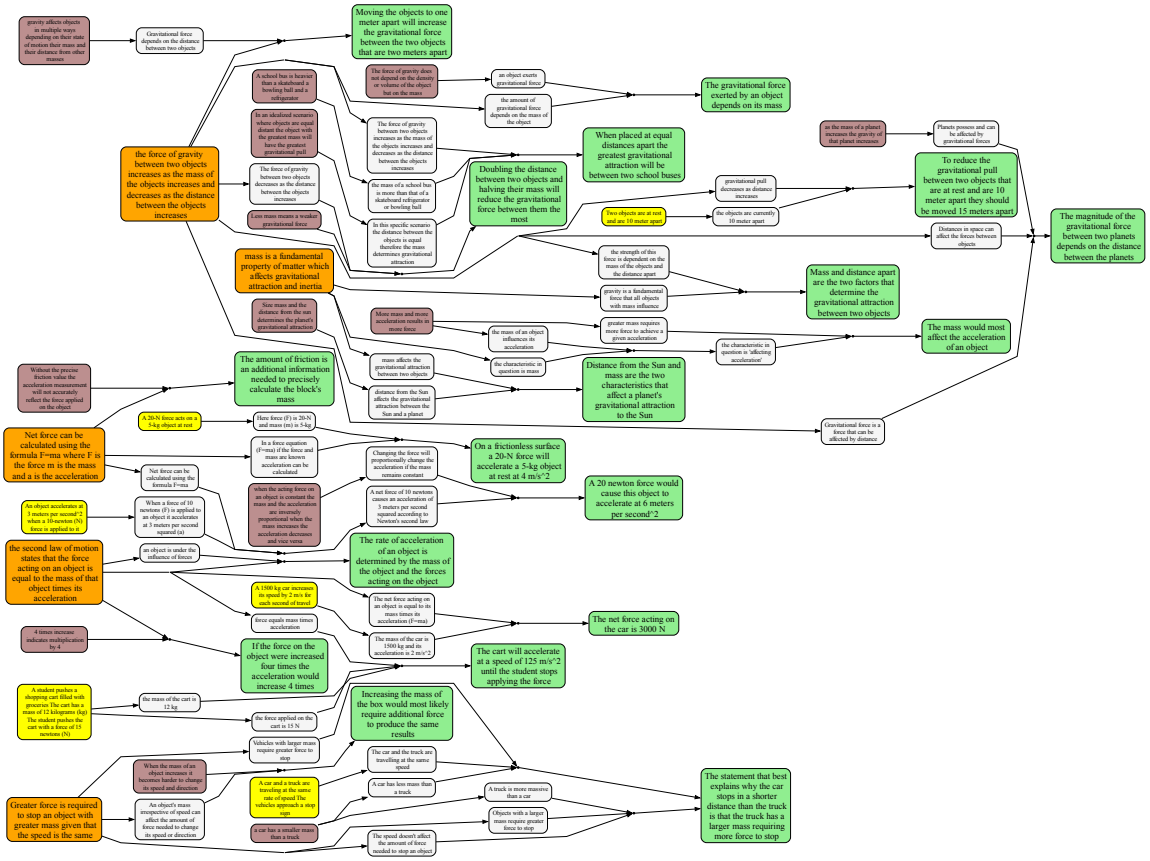


Figure 6.8: Subgraph showing some of the statements from the 200-Mt_{PC} ARC microtheory (orange) and their influence on proving training hypotheses (green) in different question contexts (yellow). The partial coverage (PC) algorithm selects facts from $\mathcal{C}$ that provide coverage for multiple questions. Facts from $\mathcal{C}$ that are specific to only one or two questions (maroon) are not selected. Each dot ($\cdot$) represents an entailment; inferences (grey and green) can be supported by alternative entailments in the form of multiple entering edges. In this instance, the PC algorithm retains 5 facts mostly related to Newton's second law, which is core to multiple subtopics in grade school physics tests (motion, friction, gravitational attraction). Only the orange nodes are retained in the microtheory for test time inference; all other statements are discarded.

Figure 6.9 shows a similar subgraph for the MedQA microtheory, this time

illustrating the facts retained by the full question coverage 100-Mt_{QC}. Hypotheses are fully thus grounded to Mt statements and the context passages of their respective questions. This graph also illustrates the effect of retaining inferences during the serial computation of the L_{kj} leaf sets.

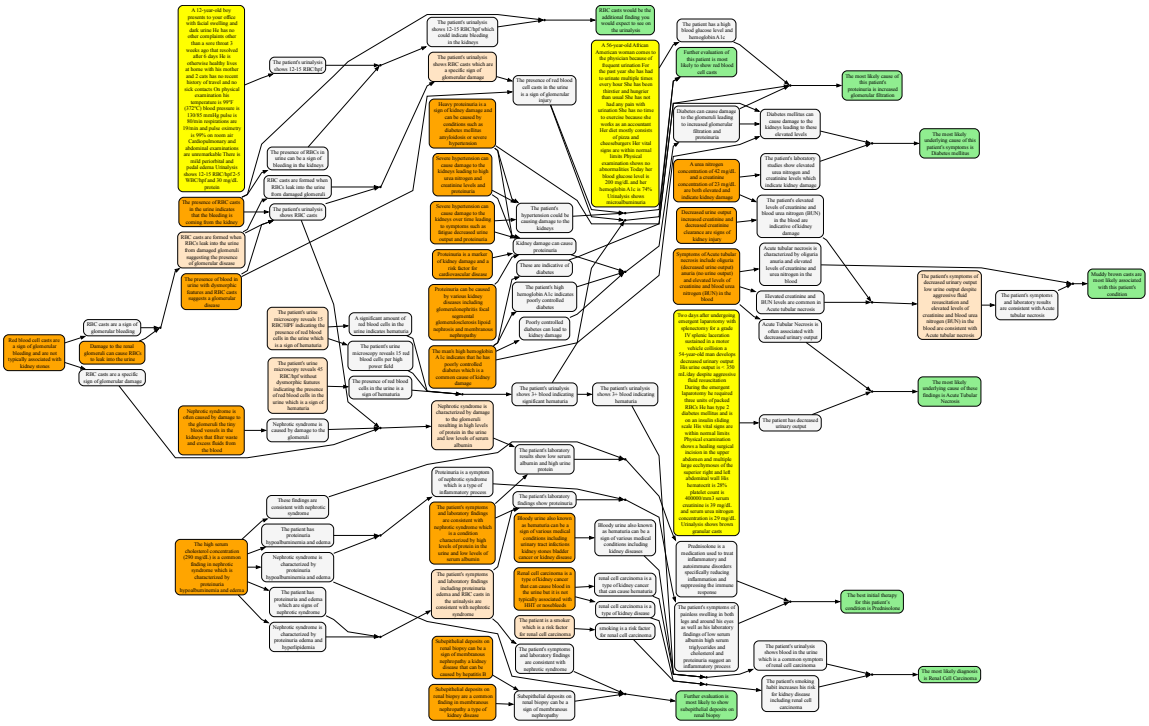


Figure 6.9: Subgraph showing some of the statements from the 100-Mt_{QC} MedQA microtheory (orange) and their influence on proving training hypotheses (green) in different question contexts (yellow) related to clinical cases. Unlike the PC approach, the question coverage (QC) approach ensures the full grounding of each training hypothesis using facts from the micro theory. As MedQA hypotheses often require many entailment hops, the inference retention mechanism described in Section 6.3.3.3 helps prove hypotheses using inferences (light orange) grounded from previous questions, some of which are omitted for space.

6.4.2.2 Coverage of Training Hypotheses

How well do the usage-based optimization methods actually do at covering the argumentative bases for the sets of training hypotheses? Figure 6.10 shows the fraction of full and partial coverage by each Mt approach. I define fractional coverage of a question Q_j as in Section 6.3.3.2: the highest fraction of leaves amongst all argumentative bases L_{kj} for the question. As expected, total coverage increases with the size of the Mt. Unexpectedly, the partial coverage linear program obtains a higher full coverage rate than the full question coverage LP; this suggests either that the latter's optimization search did not converge within the hour time limit or that it found a local minimum. The Mt_{usage} also outperforms the QC approach regarding both full and total coverage on ARC, but the trend is reversed for full coverage on MedQA. Full coverage is overall substantially lower on MedQA than ARC, highlighting that 50 to 500 facts is not enough to fully explain the distribution of questions appearing in the 4 MedQA topics comprising the data.

The coverage rates of 88.3 and 74.2 represent the effective maximum total coverage of each training set given the results of the entailment engine; it failed to find a single basis for 12% and 26% of the ARC and MedQA hypotheses, respectively. The partial coverage algorithm is, therefore, able to achieve 77-80% of the optimal coverage rate using half as many facts ($n = 500$ vs. 1000).

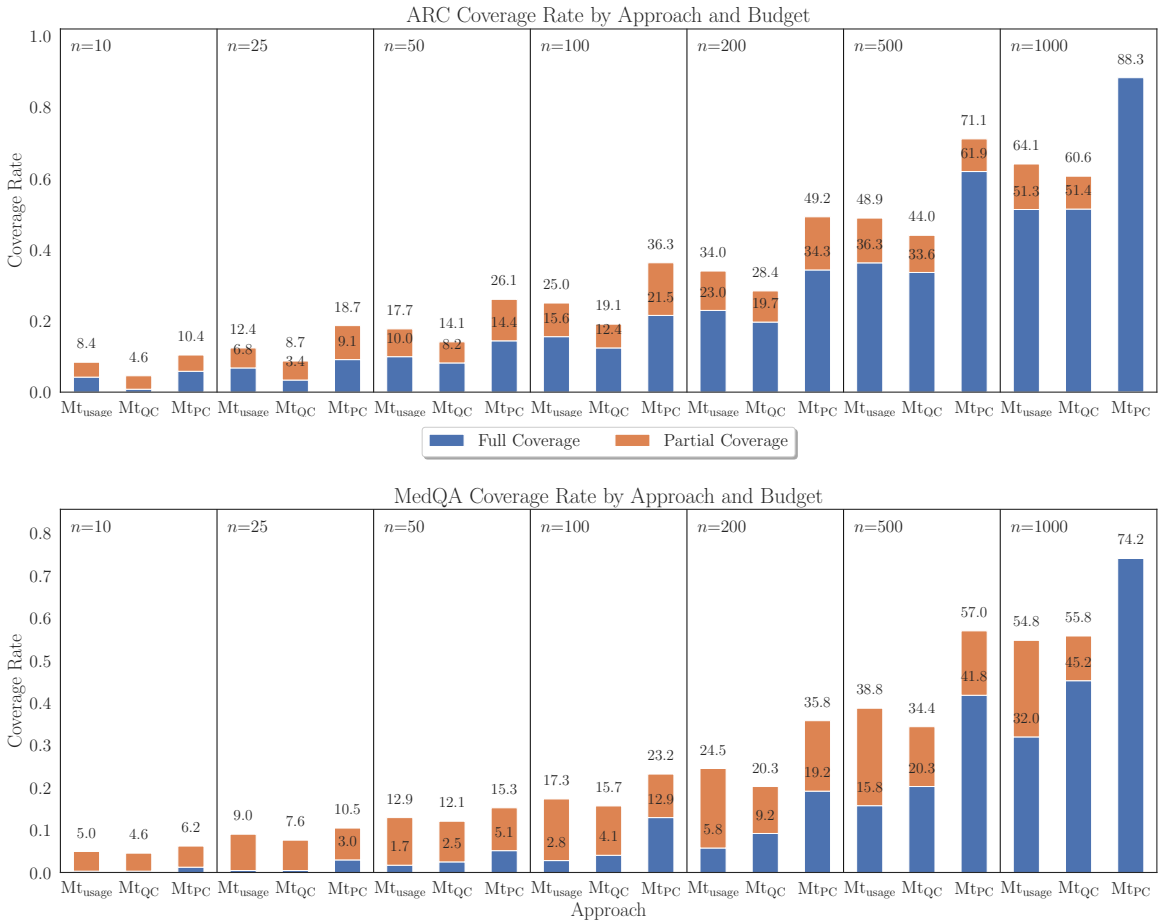


Figure 6.10: Rate of coverage of training hypotheses produced by the different microtheory approaches. Orange bars are the sum total of fractional coverages for questions not fully covered by the Mt.

I also implemented a version of the linear program that found the true optimal set of facts, i.e., the smallest set that fully covers bases for all hypotheses for which the prover found at least one. Table 6.2 shows that 744 ARC questions over 9 topic clusters required a minimum of 800 facts for "full coverage," a ratio of 1.08 facts per question. Meanwhile, the MedQA questions over 4 clusters required nearly twice as many facts per question.

Dataset	ARC	MedQA
# Topic Clusters	9	4
# Training Questions	854	641
# Questions with Proof	744 (89.3%)	476 (74.2%)
Minimum Facts for Proof Coverage	800	918
Fact/Question Ratio	1.08	1.91

Table 6.2: Results of "minimum fact" linear program that finds the fewest facts that totally cover a set of training hypotheses.

While this is a noisy approximation,[16] it highlights that the total amount of knowledge required to answer MedQA questions (in the form of generic factual statements) appears to be substantially higher for MedQA than ARC questions, which is somewhat unsurprising given the discrepancies between domains; grade school science questions test for rote memorization and application of a specific subset of general science facts by young children. The USMLE exam questions test for rote memorization of a much larger amount of information by medical students 10 to 15 years older. MedQA questions require a complex understanding of anatomy, specific drugs, and clinical decision heuristics that cover a broader scope of knowledge.

6.4.3 Test Evaluation

I run a series of evaluations in an effort to test which of the perceived benefits illustrated in Figure 6.1 of **efficiency**, **completeness**, and **seeded reasoning** show to materialize when the Mts are used to answer unseen test hypotheses.

[16]The entailment engine can produce errors and produce proofs that use fewer facts than would actually be sufficient argumentative bases for each hypothesis.

For **efficiency**, we will test whether more or less work has to be done to prove hypotheses on the basis of the number of hops per entailment tree. For **completeness**, we will examine whether we improve the rate of full test hypothesis grounding. And finally, for **completeness**, we will test how well the entailment engine performs multiple-choice question answering using the QA scenario explored in Chapter 3 and Chapter 5.

6.4.3.1 External Knowledge Sources

Entailment engines are designed to answer whether a corpus fully entails a hypothesis. During extraction, the facts in $\mathcal{F}$ and $\mathcal{C}$ are likely to fully ground a training hypothesis because they contain facts coming from a TheoryCoT LLM prompt specifically for the corresponding training question. Accordingly, I could evaluate the extent to which our microtheory methods were able to cover the argumentative bases that the entailment engine consistently found within these factpools. This is less likely the case for test questions– full grounding of test hypotheses might require more idiosyncratic, question-specific facts that are unlikely to be extracted and retained after optimization for the most generalizable facts. For example, a question about a highly specific plant breed might require the fact about that breed; if there was no training question about that breed, it is unlikely the Mt will contain the fact.

I thus propose to use the Mt in conjunction with a larger encyclopedic knowledge source. I use a retrieval index over Wikipedia (the same as used in Chapter 5) as our

base corpus and then evaluate the impact of combining that index with a microtheory.

For MedQA, I also add an index over the medical textbooks released as a support knowledge source with that dataset. Whenever TreeWise invokes its document retrieval module for a given hypothesis, I concatenate the results of retrieving from the Microtheory to the results from the external sources, thereby doubling the number of support documents checked for entailment of a query hypothesis and, in the absence of an identified entailing document, the number of documents shown to the forward-chaining LLM prompt when producing candidates decompositions of the hypothesis.

6.4.3.2 Baselines

To illustrate the effectiveness of the optimization methods, I compare them against a baseline that samples n random facts from $\mathcal{F}$. I also compare against using the entire raw pool $\mathcal{F}$ as the Mt. For ARC, I compare against the gold WorldTree resource, the version of which I use has 12,657 facts. I also consider using only the facts in WorldTree labeled as "core" by the WorldTree annotators for at least one question in their dataset (8722 of 12657 facts).

6.4.4 Coverage of Correct Test Hypotheses

I first evaluate whether microtheories improve the engine's ability to entail true test hypotheses (declarativized correct answers to unseen test questions) using the combination of a microtheory and an external corpus. Figure 6.11 depicts the

hypothesis grounding rates resulting from adding various microtheories as additional sources of grounding complementing an external corpus (Wikipedia for ARC, Wikipedia and the MedQA textbooks for MedQA; labeled 'Corpus' in the figure).

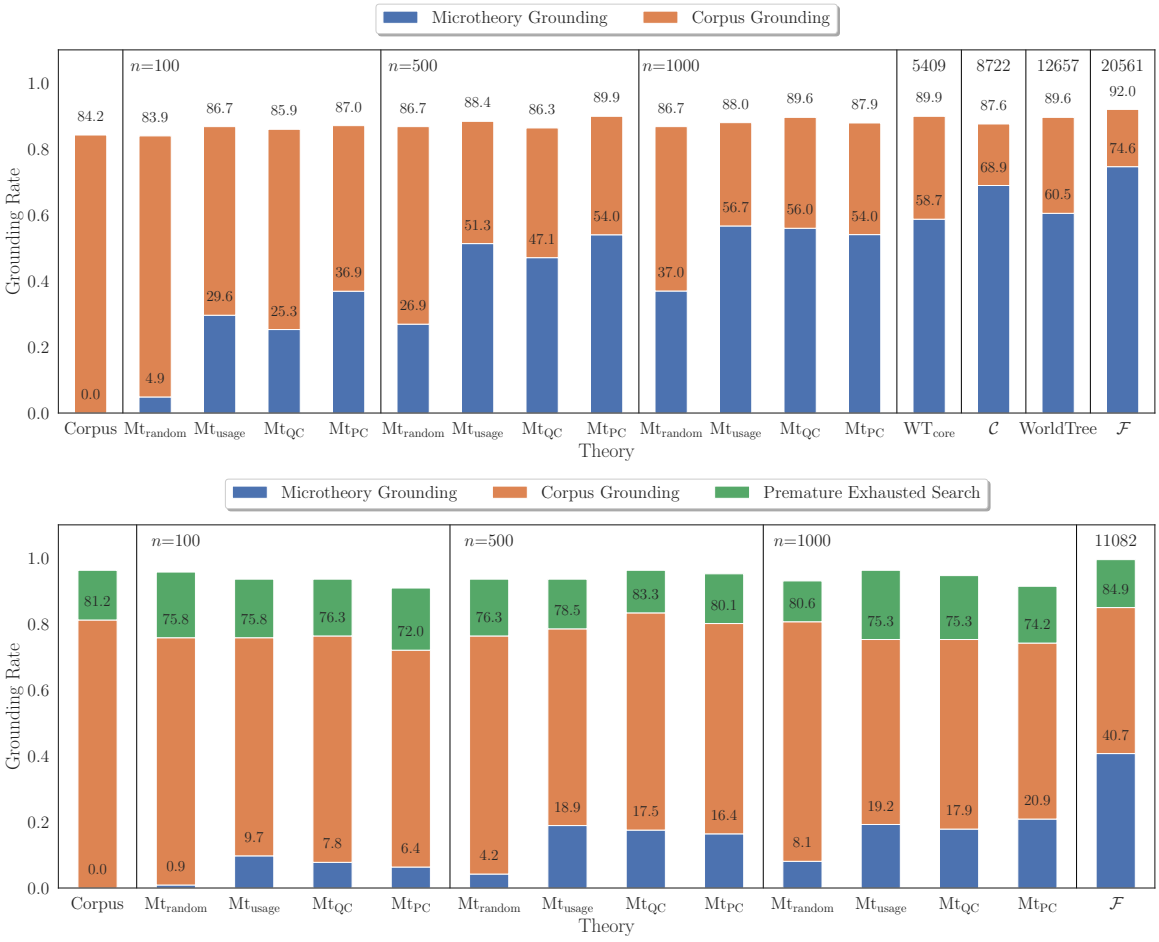


Figure 6.11: Effect on ARC (upper) and MedQA (lower) test hypothesis grounding rate by adding microtheory approaches as additional grounding premises to an external corpus.

I have broken down the overall coverage rate (the fraction of hypotheses for which it finds a proof) into two components: the fractions of found trees whose leaves are grounded in the Mt (blue) and the fraction grounded to passages from the external

corpus (orange).[17] For ARC, there is a clear relationship between Microtheory size and an increase in both the overall hypothesis coverage rate (blue+orange) and rate of grounding to the Mt. At the extreme of using the entire fact pool $\mathcal{F}$, the engine proves 8% more hypotheses and grounds them to the pool at a rate of 74.6%. This is a higher overall grounding rate than using WorldTree, which purports to contain the core facts necessary for all ARC questions (it does not contain all esoteric facts, e.g., "n ibex is a mammal"). Using the "core" facts from WorldTree obtains the same coverage rates as using the full knowledge store. The optimization methods show a clear trend at lower Mt sizes (100,500); the *partial coverage* Mt shows higher grounding than *usage* then *question coverage*, in that order. All outperform the random baseline.

For MedQA, the Mt grounding rate (blue) increases with Mt size, but the relationship between overall grounding and size is less clear, though $\mathcal{F}$ still obtains the highest overall rate. Notably, the rate *decreases* when adding all non-$\mathcal{F}$ theories compared to only using the Corpus of Wikipedia and textbooks. This contradicts a running assumption that "more information is always better"; something about the added Mt information impacts the prover negatively. One possible reason for this is that the Mts do not contain relevant or useful information for some subset of the questions. The engine generates a fixed number of decompositions conditioned on retrieved content; if I use multiple sources of knowledge (the Corpus AND a microtheory), I concatenate the content retrieved separately from each source and feed

[17] If the prover finds multiple trees, I take the one with the highest fraction of microtheory-rooted leaves.

it to the decomposition-generating LLM prompt. Whether the Mts contain relevant information for the test questions is further explored below in Section 6.5.

If the retrieved information from the Mt is not useful, this generation step might generate valid and groundable decompositions at a lower rate, thus decreasing the overall grounding rate. To partially illustrate this, I have included the rate of *premature search exhaustion* (green) to the MedQA graph in Figure 6.11. This refers to cases in which the prover fails to find a proof not because it hits the predefined expansion budget (80), but runs out of search branches to consider because it has ruled all proposed decompositions of the hypothesis to be invalid entailments. This rate often increases when using an Mt versus only using the external corpus.

6.4.4.1 Effect of Mts on Proof Depth

One of my argued hypotheses for why Mts should benefit from the entailment engine's performance was that they amortize inference by saving useful statements that, therefore, wouldn't need to be re-proved at test time. This would be shown at test time via the average depth of entailment trees produced by the engine with vs without a Mt available. Figure 6.12 shows the average depth of the *smallest* proof tree found by the engine for each test hypothesis in the ARC topic-specific test subset. Average depths with or without Mts all lie between 1.8 and 2.1, without any clear relationship between approach or size and resulting depth. I therefore don't have evidence that the hypothesis about amortizing deeper searches is true for this dataset and scenario.

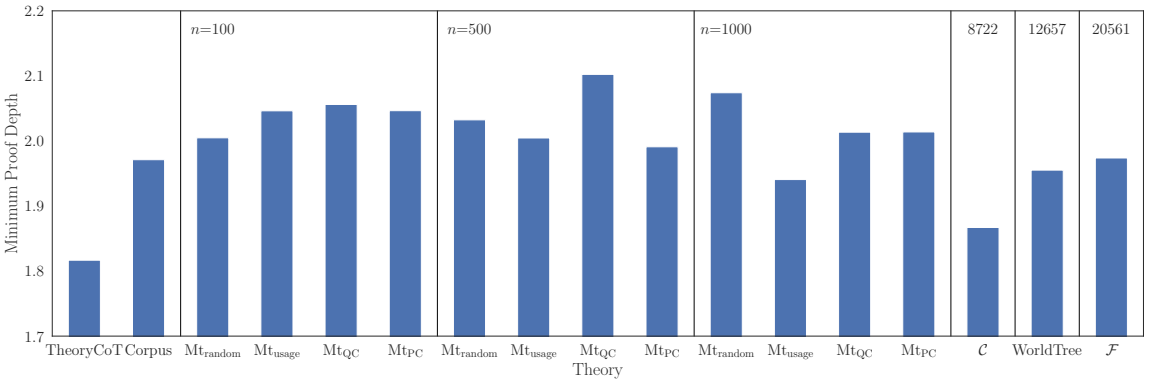


Figure 6.12: Effect on ARC hypothesis proof depth by adding various microtheory approaches to Wikipedia.

6.4.4.2 Effect on Per-Fact Usage

I have described various ways to extract a bundle of the most core, reusable facts for a given domain based on a set of training questions. One way to partially measure how well we have achieved this is to track how frequently the average fact is reused on test questions. Intuitively, one would want all facts in the Mt to be frequently reused and for none of the facts to *never* be used. Figure 6.13 depicts the average number of questions per fact in various Mts (upper) as well as the fraction of the Mt that is never used for a single test question (lower).

We can see that there is a natural inverse relationship between Mt size and per-fact usage rate, with 100-sized Mts having facts that are used on 2-3 questions on average, but 1000-fact Mts are used for less than 1. The *partial coverage* microtheories have slightly lower usage rates and slightly higher unused fact rates than the other optimization methods. Below I show that the *partial coverage* Mts are *better at*

entailing training hypotheses (Figure 6.10), suggesting that they have perhaps overfit to the training questions and captured idiosyncractic facts that are less reuseable for new questions. The full fact pool $\mathcal{F}$ and condensed fact pool $\mathcal{C}$ have very low mean usage and very high amounts of used facts, while showing in experiments below to improve hypothesis grounding and question answering *more* than the budgeted smaller Mts.

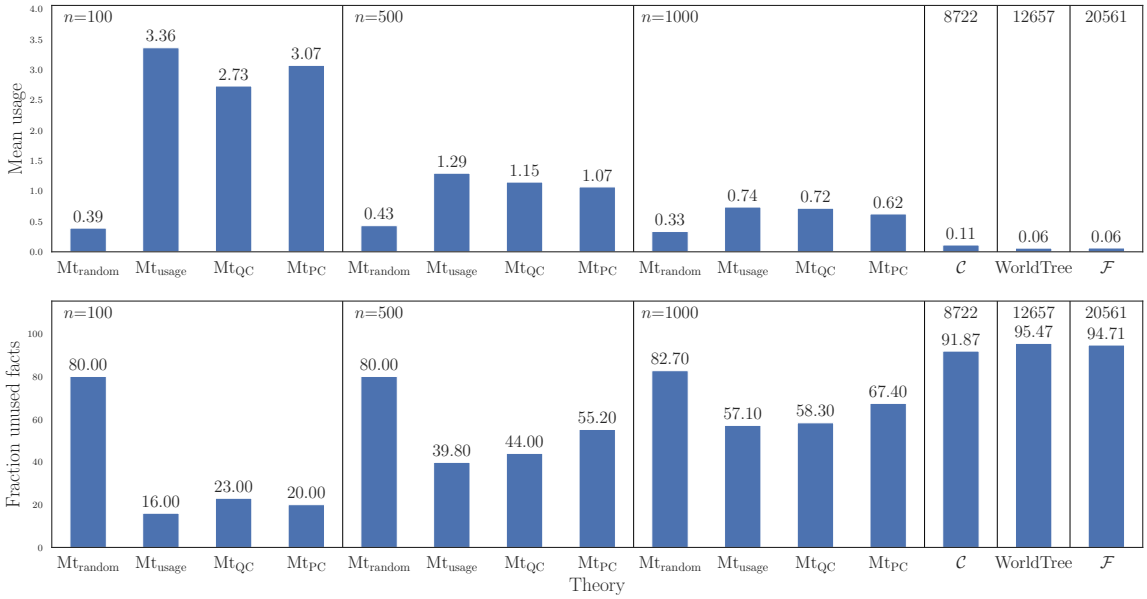


Figure 6.13: (Upper) Rate of average per-question fact usage in each ARC microtheory. (Lower) Fraction of each ARC microtheory not used for any proof.

6.4.5 Entailment Engine Question Answering

6.4.5.1 ARC QA Results

Figure 6.14 shows the results of running multiple-choice question answering on the 9 ARC topics using TreeWise to prove each answer option and taking the highest scoring tree as the predicted answer. Using only the Wikipedia index as the knowledge

source (Corpus, leftmost bar) achieves 69.2% QA accuracy. Adding 100 or 500 facts with any approach generally does not improve this and can decrease it using 100 random facts.

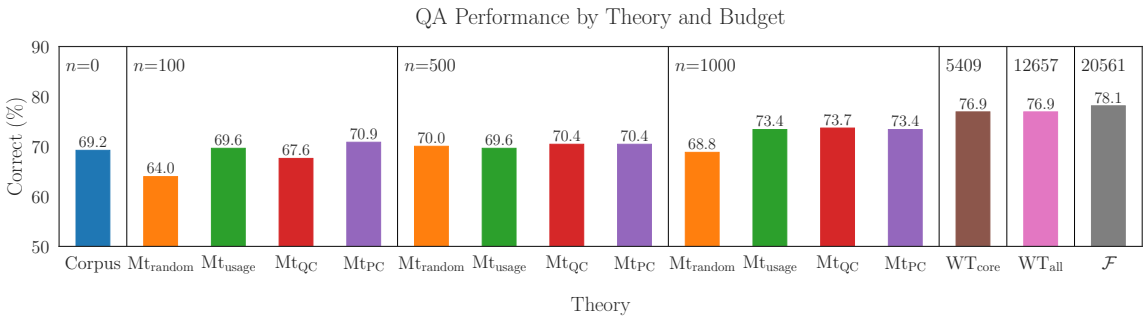


Figure 6.14: Question Answering performance by TreeWise engine on ARC subset using Wikipedia plus various microtheories as knowledge sources.

When adding 1000-fact theories, all optimization methods add 4 points in QA accuracy (73%). Adding the entire fact pool $\mathcal{F}$ further improves performance by 5 more points (78%). This exceeds the performance gains of using WorldTree, either the whole corpus (WT_{all}) or just its core facts (WT_{core}), which both add 8 points (77%). These results suggest the following conclusions:

- Size correlates with QA accuracy, as it does with true hypothesis grounding as shown in Section 6.4.4. Extracting core facts and dropping the less frequently used ones decreases performance compared to using the full pool. For the subset of ARC considered, one needs to retain at least 500 facts to see a noticeable improvement in QA accuracy.

- The question-driven Mt extraction method can provide performance gains on

par with a clean, hand-annotated corpus (WorldTree).

- When extracting core facts from $\mathcal{F}$, the choice of optimization method doesn't have a noticeable effect on QA accuracy. Each method outperforms randomly selecting a set of facts of the same budget size.

6.4.5.2 MedQA Results

QA results are more mixed for MedQA. The overall hypothesis grounding rate is higher when using $\mathcal{F}$ for both domains (green + yellow + purple bars), but the QA accuracy (green + yellow) is lower for MedQA because more proved incorrect options outscore correct ones (purple).

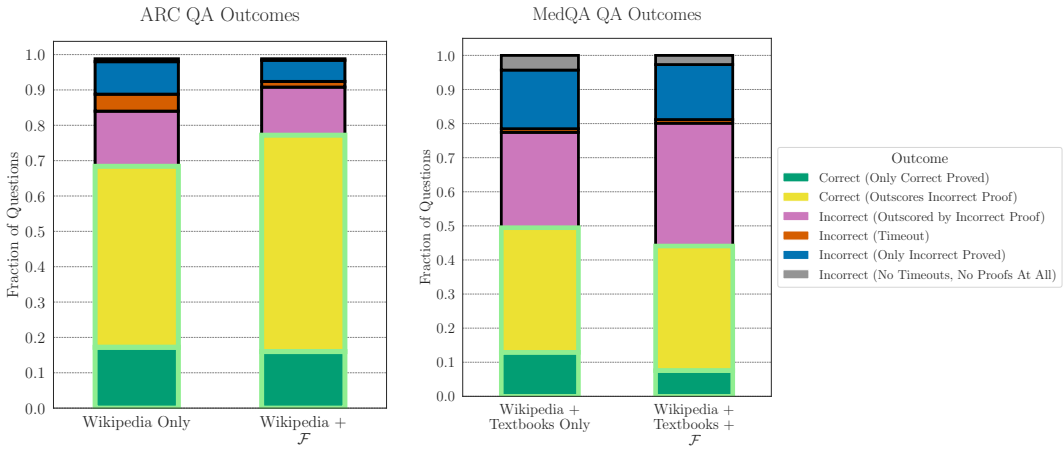


Figure 6.15: Breakdown of QA runs by outcome (correct/incorrect answer proof results). The rate at which only the correct answer is proved decreases when adding $\mathcal{F}$, as does the overall rate of proving the correct answer. However, this does not correspond to higher QA accuracy, as the rate at which both the correct and at least one incorrect are proved is higher. The rate at which the correct proof outscores all incorrect increases only for ARC but not MedQA; the rate at which an incorrect answer outscores the correct is substantial higher for MedQA, leading to lower overall QA accuracy when using $\mathcal{F}$.

These results show that an improved grounding rate is not a silver bullet for improved QA performance, especially for challenging medical questions. USMLE questions require integration of granular domain knowledge (more facts on average than ARC, as shown in Table 6.2) with critical reasoning and medical heuristics that are difficult to learn for adult medical students. The insertion of some relevant core facts does not guarantee that they will be applied in the correct fashion in order to answer hard questions.

6.5 Relevance to Test Questions

Intuitively, a microtheory should contain generalizable, useful statements that help solve any question on a given topic. In this project, I have proposed a *question-driven* approach to constructing a microtheory, which thus relies on the assumption that the kernels of knowledge needed to answer a set of training questions should be predominantly the same kernels needed to solve test questions. If answering a given MedQA test question correctly relies on knowing about a particular drug, but none of the training questions required knowing that drug, the Mt likely won't have any facts about it. Given the partial negative result depicted in Figure 6.11 in which adding Mts *decreased* hypothesis grounding rates rather than increasing them, I explored whether this can partly be explained by their failure to contain relevant facts for test questions.

6.5.1 Relevance Rating Experimental Setup

I constructed an LLM-based metric that lower-bounds an Mt's relevance to a given question by checking whether it contains *at least one* fact that is directly relevant to getting the question right, i.e., differentiating the correct answer from any incorrect options. For each question, I used the TreeWise support fact dense retrieval index to obtain some top k=270 facts relevant to the correct answer. I then prompted GPT-4[18] to rate each retrieved fact on an ordinal relevance scale based on the annotation rubric used by Jansen et al. (2021) to score the relevance of WorldTree facts using human experts. The prompt used for scoring facts is shown in Figure 6.16. The original rubric by Jansen et al. (2021) contains 4 ordinal scores; I added a 5th to differentiate between facts relevant to ruling out incorrect options (4) vs supporting correct options (5). I then measured the rate at which at least one 5-rated fact appeared in the retrieved fact set, using that as a soft indicator that the Mt contains any useful information at all that would be core to an argumentative basis for the correct hypothesis option. This is almost certainly *not* counting whether the Mt contains enough knowledge to fully explain correct answers, which more closely resembles the functionality of the entailment engine.

Figure 6.17 shows the change in per-question relevance rate as I increase Mt size and vary the selection methods. As expected from the grounding rate results above, the ARC Mts contain 5/5-relevant facts at a much higher rate than MedQA. The

[18] `gpt-4o-2024-08-06`

```
You are a scientific expert that helps other scientists determine whether various facts are relevant to explaining the correct answer to
      a question. Scientists give you a question and a list of facts that they believe are relevant to the question. Your job is to
      determine whether each fact is relevant to explaining the correct answer to the question using a provided rubric for scoring:

A fact has a score of 5 if it has the following indicators:
* core/critical fact/knowledge that explains why a correct answer is correct
* essential piece of information if explaining phenomenon in question to a toddler

A fact has a score of 4 if it has the following indicators:
* core/critical fact/knowledge that explains why an incorrect answer is incorrect

A fact has a score of 3 if it has the following indicators:
* moderately important fact necessary in a correct explanation but not being explicitly tested *underlying assumption necessary to
      understanding the terms used in the correct response and explanation
* defines terms used in the prompt and/or the correct answer
* semantic statements: correctly identify a term in the immediate question or correct answer
* common sense statement that is topically relevant but not necessary for differentiating correct from incorrect

A fact has a score of 2 if it has the following indicators:
* extra detail - explanations missing this fact are neither incorrect nor incomplete
* true relevant statement but not necessary for understanding phenomena in question

A fact has a score of 1 if it has the following indicators:
* irrelevant
* incorrect

You do NOT need to give a 5 for at least one fact. You should give a 5 only if the fact is absolutely necessary and core to correctly
      answer the immediate question (i.e. if someone does not know the fact, they cannot answer the question correctly).

Here are some of the questions that you ask yourself when considering a fact:
Does this statement directly answer the original question? (yes = 5)
* Is the information absolutely necessary to correctly answer the immediate question? (yes = 5)
* Is the information absolutely necessary to explain why another answer is incorrect for the immediate question? (yes = 4)
* Is the information important to understanding the correct answer or any of the key terms being tested by the immediate question or
      correct answer (e.g., definitions, etc.) but not the concept being tested itself? (yes = 3)
* Is the statement a semantic claim that correctly identifies a key term in the immediate question or correct answer in a way that is
      relevant to the phenomenon being tested? (yes = 3)
* Is the statement relevant to a correct explanation of the correct answer but not required for a complete and correct explanation? (yes
      = 2)
* Does the statement add correct and relevant supporting details that give context or depth to an explanation but are not required to
      make the explanation complete and correct? (yes = 2)
* Is the statement a semantic claim that correctly identifies a term that is related to the immediate question or correct answer but not
      required to produce the correct answer or explain it? (yes = 2)
* Is the statement completely irrelevant to a correct and complete explanation of the correct answer? (yes = 1)
* Is the statement incorrect? (yes = 1)
```

Figure 6.16: Prompt used to score the question relevance of retrieved microtheory statements. Instructions are based on the rubric from Jansen et al. (2021).

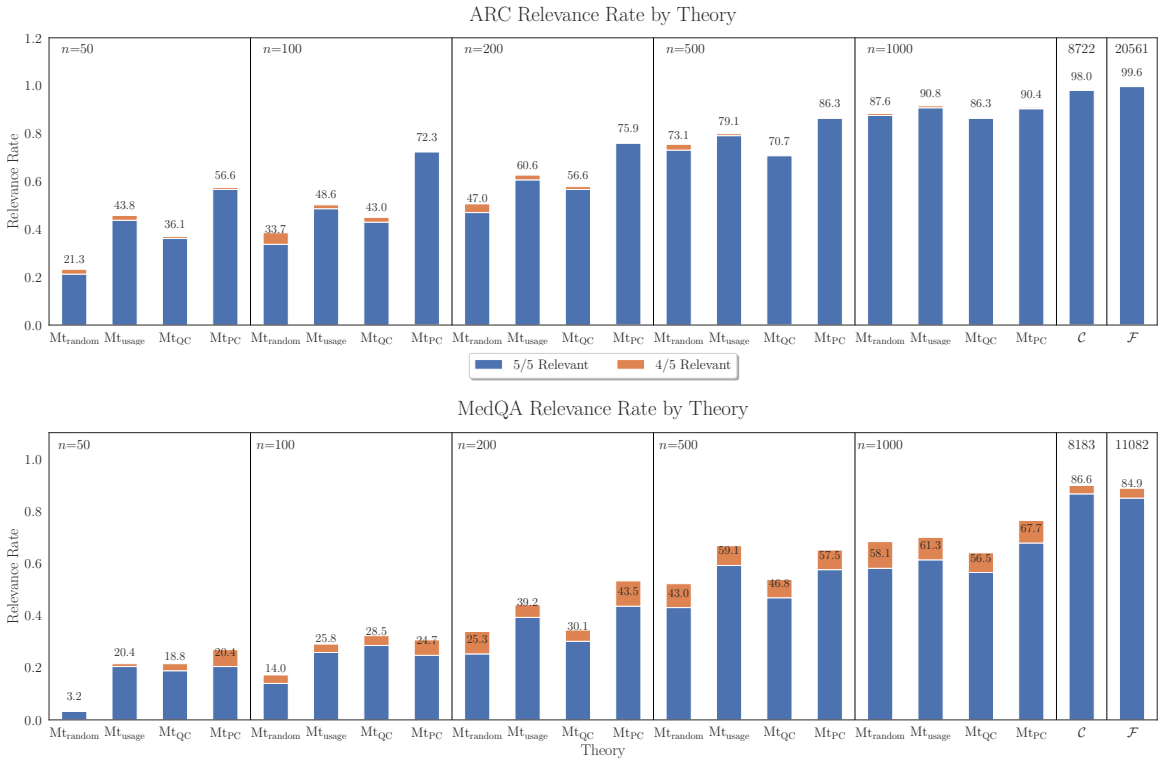


Figure 6.17: Rate (%) of microtheories containing at least one fact with 5/5 and 4/5 relevance as determined by GPT-4o. The numbers above each bar represent the percentage of 5s.

best-performing optimization method, *partial-coverage*, obtains a 56% relevance rate for ARC with only 50 facts and a 72.3% rate with 100. For MedQA, the best rates go down to 20.4% and 28.5%. At 1000 facts, all Mts have around 90% relevance for ARC but only 55-67% for MedQA. The fact pool $\mathcal{F}$ and condensed pool $\mathcal{C}$ are nearly 100% relevant for ARC but only 85% for MedQA.[19]

This provides a possible explanation for the ineffectiveness of the MedQA Mts: at best, the largest 1000-fact-or-less Mts contain at least one relevant fact only 2/3 of the time. If the entailment engine is provided with Mt facts during decompositional search, they are likely not helpful in discerning correct answer hypotheses.

To illustrate this, I examined the difference in performance between the entailment engine run using the 1000-Mt_{PC} versus using only the Corpus on the MedQA questions. The Mt contains a relevant fact for 126/186 test questions. For these questions, using the Mt in addition to the corpus achieves a grounding rate 1% *lower* than not using it (not statistically significant). However, for the 60 questions for which the Mt does *not* contain a relevant fact, using the Mt drops the grounding rate **by 20%**. I examined other microtheories of different sizes and approaches and found the consistent trend that performance essentially stayed the same for the 126 relevant-fact questions but dropped universally on the 60 no-relevant-fact questions (albeit by 10% at times rather than 20). For the raw fact pool $\mathcal{F}$, for which the overall grounding rate is 84.9% versus 81.2% using the corpus alone, grounding increases by 8% on the relevant-fact

[19]That the rate drops by a point between $\mathcal{C}$ and its superset $\mathcal{F}$ is likely a result of retrieval issues, i.e. failing to find a 5/5 fact in the top-270 facts in the larger and noisier $\mathcal{F}$.

questions but decreases by 5% on the no-relevant-fact. This suggests strong evidence that the drop in grounding performance when adding the Mts correlates with a lack of relevant information for the question.

6.6 Expert Relevance Annotations

In Section 6.5, I found that Mts created for MedQA questions were, on average, less relevant than ARC Mts for test questions in the same topic clusters as the training questions used to create the theories. In this section, we explore how useful human domain experts find the facts in each theory. I recruited two late-stage medical students from Thomas Jefferson University to annotate the general relevance facts from different MedQA microtheories. I worked with the annotators to develop a 5-point rubric for scoring how relevant a fact is to the core subjects that appear on the USMLE. The rubric is shown in Figure 6.18 and Figure 6.19.

I sampled 25 facts (fewer when necessary) from Mts with 70 budgets of size ranging from 10 to 1000. I evaluated the Mt_{usage}, Mt_{PC}, and $\text{Mt}_{\text{random}}$ approaches. Both annotators rated each fact independently and then reconciled the (rare) disagreements. I computed agreement metrics across the full set of 314[20] annotated facts, finding a Krippendorff α of .81, Cohen's κ of .809, and raw agreement rate of .863.

Average relevance ratings per microtheory are shown in Figure 6.20. Both

[20]This number is not $7 \times 3 \times 25$ because some Mts were smaller than 25 facts, the n-Mt_{usage}s are all supersets of each other, and many other samples contained high overlap due to the extraction algorithm behavior.

#	Indicators	Exemplars
5	• [1] Statement/fact that includes relevant associations* and **IS** specific, comprehensive, or pathognomonic enough to provide a clear answer to a test question. • [2] Statement/fact that provides **core** or **comprehensive** information and/or associations about a topic domain** that are **fundamental** to its understanding as a whole	*[1] Deletion of 11-p-15 can cause Beckwith-Wiedemann syndrome, which includes symptoms such as macroglossia, omphalocele, and hemihypertrophy* *[2] Tachycardia (elevated heart rate) and tachypnea (elevated respiratory rate) are signs of cardiac or pulmonary distress.*
4	• [1] Statement/fact that includes relevant associations but **is NOT** specific, comprehensive, or pathognomonic enough to provide a clear answer to a test question • [2] Statement/fact that provides **neither core nor comprehensive** information and/or associations about a topic domain that are relevant but **not fundamental** to its understanding as a whole ○ Without knowing this statement/fact, it **would** be difficult to correctly answer a question on this topic • [3] Definition that **includes** relevant associations, context, and/or assumptions that are directly useful in answering a question correctly	*[1] In autosomal recessive inheritance, both parents must be carriers of the mutated gene for a child to inherit the disease* *[2] Increased circulating estrogen can occur in men with liver disease, such as alcoholic cirrhosis* *[3] Osteoporosis is a condition characterized by low bone mass and deterioration of bone tissue, leading to increased bone fragility and risk of fractures.*
3	• [1] Statement/fact that is accurate **and** provides context or information about a topic; **does not** provide information that would directly result in getting a question right. ○ Without knowing this statement/fact, it **would NOT** be difficult to correctly answer a question on this topic • [2] Definition that **lacks** relevant associations, context, and/or assumptions that are directly useful in answering a question correctly ○ May include examples or additional details that do not directly facilitate a correct answer	*[1] Symptoms of drug-induced fever include fever and confusion* *[2] A Physician Health Program is a program designed to help physicians with substance abuse or mental health issues*
2	• [1] Statement/fact that is **generic** or **common-sense;** information is widely known and/or lacks details or relevance about a given topic • [2] Statement/fact that **rarely** helps in the specific context of medical exams, offering **little** value in the diagnostic and/or question-answering process • Statement/fact that contains an **element or component** that is **factually untrue** or **inaccurate**	*[1] The spine is a part of the musculoskeletal system* *[2] Two-dimensional gel electrophoresis is not typically used to diagnose genetic disorders* *[3] Decreased activity of epithelial Na+ channels in principal cells can be caused by increased activity of luminal K+ channels in principal cells*
1	• [1] Statement/fact that is **useless**, **unclear**, and/or completely **irrelevant** • [2] Statement/fact that **never** helps in the specific context of medical exams, offering **no** value in the diagnostic and/or question-answering process • Statement/fact that is **entirely** factually untrue or inaccurate	*[1] The patient's BUN and creatinine levels are slightly elevated.* *[2] The cause of erythema nodosum is often unknown, or may be related to certain systemic diseases.* *[3] Bloody diarrhea can be a symptom of Giardia lamblia infection*

*Relevant association: A significant connection or correlation between two or more related factors (e.g., diseases, symptoms, or risk factors) that directly impacts the understanding, pathophysiology, diagnosis, treatment, and/or prevention of a medical condition.

**Topic domains: A core subject, discipline, or system in medicine that encompasses many which many questions are derived from (e.g. cardiac system, renal system)

Figure 6.18: Rubric used to collect expert relevance ratings for medical microtheory facts (1 of 2)

- Would a medical student be in trouble if they didn't know this fact the morning of the exam? (yes = 5)
- Can you imagine up (or remember) a question for which this statement would directly answer the question? (yes = 5)
- Would a topic domain (e.g., "cardiovascular system") be hard to understand if you didn't know this fact? (yes = 5)
- Could the information feasibly be important to understanding the correct answer or any of the key terms being tested by questions or correct answers (e.g., definitions, etc.) but not the concept being tested itself? (yes = 4)
- Does the statement add correct and relevant supporting details that give context or depth to an explanation, facilitating understanding of the concept? (yes = 4)
- Is the statement relevant to a correct explanation of a correct answer but generally not required for a complete and correct explanation? (yes = 3)
- Is the statement a semantic claim that correctly identifies a term that might be related to questions or correct answers but not required to produce the correct answer or explain it? (yes = 3)
- Is the statement partially incorrect or contains an element of untruth? (yes = 2)
- Is the statement "common knowledge"? (yes = 2)
- Does the statement have information that is too general or is only marginally related to a given topic? (yes = 2)
- Is the statement completely irrelevant to a correct and complete explanation of any feasible correct answer? (yes = 1)
- Is the statement completely incorrect? (yes = 1)

Figure 6.19: Rubric used to collect expert relevance ratings for medical microtheory facts (2 of 2)

optimization approaches outperform a random sampling baseline by at least half a point. The *partial coverage* approach is consistently .6-1 point higher. It also exhibits a slight trend in which the average score decreases slightly as I increase the budget but does not drop below a 4 out of 5 score. The annotation rubric defines a 4/5 score as a relevant but either unspecific and/or not fundamental fact with respect to expected exam questions or a definition that is directly useful for answering questions (definitions are typically not comprehensive enough to be a 5). In contrast, a randomly selected fact from $\mathcal{F}$ scores lower than 3.5, indicating contextualizing information that does not directly contribute to getting questions right. These findings suggest that the Mt extraction methods are nontrivially effective at extracting the core, useful knowledge from the larger fact pool, as determined by domain experts.

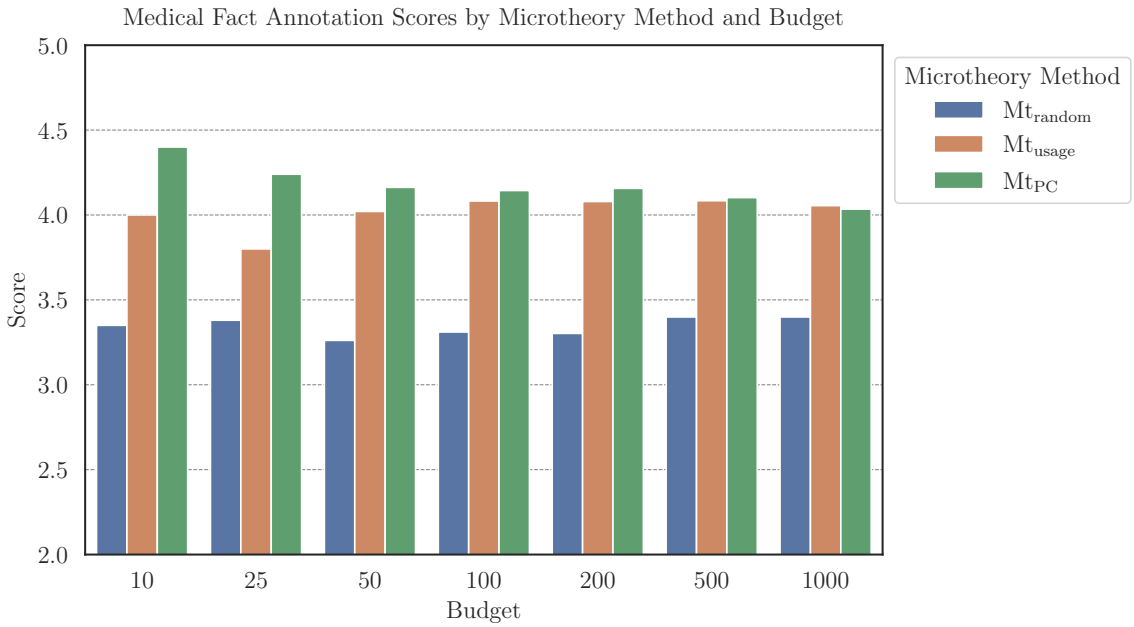


Figure 6.20: Average relevance score for microtheory facts as annotated by a pair of medical students.

6.7 Related Work

This is the first work to my knowledge that attempts to reconcile (or, at least, consider the potential benefits of) the 90s notion of symbolic microtheories with modern large language model-based reasoning, though various works have considered modern (pre-neural)implementations of microtheory-based reasoning schemes (Forbus, 2010). However, various other works have explored ways of extracting generalizable abstractions from LLMs, and of using these abstractions to improve the performance of reasoning systems on new problems.

The discipline of Inductive Logic Programming (ILP) (Muggleton, 1991) seeks to automatically learn logic programs (i.e. first-order clausal theories of facts and rules) from examples, and some have attempted to use ILP principles to learn symbolic rules from natural language (Yang and Deng, 2021).

Various computational cognitive science works have similarly considered cognitively-inspired models of concept learning and generalization, which frequently takes the form of learning symbolic facts and rules from examples through various forms of Bayesian inference (Goodman et al., 2014; Piantadosi et al., 2016). The symbolic formalism of these models is often referred to as a "Language of Thought." The output of Bayesian "library learning" approaches (Ellis et al., 2018; Ellis et al., 2021) is a set of symbolic program components meant to efficiently solve new tasks in a given domain. In contrast to these works, the microtheories I introduce in this chapter are meant to capture generalizable facts from a large language model completely in natural language but

ultimately serve a similar purpose: being able to ground new inferences via search-based reasoning effectively and efficiently using a finite, discrete set of knowledge fragments.

The question-driven microtheory pipeline is related to other attempts to use LMs to construct indices of silver-quality (i.e. model-annotated) task knowledge. The Probably Asked Questions (PAQ) resource is another approach that starts from a set of QA examples and uses an LM to generate a dataset that can be used to improve the performance of QA systems on new questions (Lewis et al., 2021). In their case, the LM is trained on QA data to generate the question from the answer and is used to generate new questions that are likely to be asked about the lines in a large text corpus like Wikipedia. The TeachMe system (Dalvi Mishra et al., 2022) is another example of a question-driven methodology that saves an explicit memory of knowledge statements, a version of "user feedback memories" also explored by **tandon-etal-2022-memory**. CLIN (Majumder et al., 2023) is another example of automatically learning abstractable statements about a task (in their case, a grounded text world environment) based on the task's training data.

6.8 Discussion

In this chapter I explored the automatic generation of entailment-supporting microtheories, which are small, domain-specific knowledge buffers that can be used to improve the performance of entailment engines on new domains. Entailment engines

determine whether hypotheses are supported by some argumentative basis stored in an external corpus of knowledge; the microtheories I introduced in this chapter are a means to automatically augment a buffer of explicit knowledge that can be used to support new entailment decisions. I introduced a series of question-driven mechanisms (§6.3)for extracting generalizable pieces of domain knowledge from LLMs and identifying the subsets of core knowledge most likely to be reused in a given domain defined by a dataset of training questions and corresponding hypotheses (§6.3.3). These methods leverage the entailment engine framework developed in previous chapters to search for supporting bases for hypotheses in a corpus of knowledge, which are then used to distill the n most useful and generalizable facts from that corpus to form a microtheory.

I showed through qualitative and quantitative experimentation (§6.4) that these automatically extracted microtheories contain predominantly generalizable and useful facts in multiple problem domains, and that providing them to an entailment engine at test time can improve the engine's performance on new questions in those domains– both in terms of the ability to ground new hypotheses (§6.4.4), and (for certain domains) the ability to answer multiple-choice questions correctly (§6.4.5). Expectedly, I observed that larger microtheories are capable of grounding more hypotheses, but that the various techniques I introduced for distilling the microtheories can help to mitigate the noise introduced by the repetition and semantic equivalence of facts at the cost of some coverage.

The results of this chapter suggest that the automatic generation of microtheories

can be a useful tool for improving the performance of entailment engines on new domains, though we also observed that the utility of the microtheories can vary depending on the domain and the extent to which the knowledge relevant to a set of training questions is likely to "cover" the knowledge required to answer new questions in the same topic (§6.5). While a question-driven methodology for extracting microtheories has shown to be very effective for e.g. grade-school science, it may not be as effective for more complex domains such as medical question answering, where the knowledge required to answer questions is more granular, diverse, and specialized.

6.8.1 Limitations and Future Work

This work considered two domains of knowledge-centric question answering: grade-school science and medical question answering. In the former domain, we observed a high degree of success at automatically constructing a microtheory to cover 9 frequent high-level topics in the dataset. For a set of 4 topics in the latter domain, we observed that the microtheory was less effective at improving the performance of the entailment engine; analysis suggested that this was due to the more specialized and diverse nature of the knowledge required to answer medical questions. While I provided some exploration of this shortcoming in the medical domain, future work could explore more deeply the reasons why the microtheory was less effective for medical QA, and reconsider what types of knowledge might be more useful for improving the performance of entailment engine. I observed these shortcomings for specific subsets of

questions in two domains selected specifically for topical similarity; future work might consider whether larger-scale training datasets—both in total quantity of questions and in topics covered–might be able to provide more generalizable knowledge for a wider range of questions in a domain.

Future work could also explore the effectiveness of microtheories in other domains, such as history, law, or other scientific disciplines, to determine the extent to which the question-driven methodology for extracting microtheories is generalizable across different types of knowledge. However, the differing levels of success in the two domains considered in this work suggest that there is a spectrum of domains for which the microtheory knowledge structure, as I have conceptualized it in this work, may be more or less effective. While ideally, the microtheory should be universally beneficial for entailment engines in any new problem domain, the results on the medical domain suggest that this may not be the case, and that the effectiveness of microtheories may be contingent on the nature of the problem and the breadth and flavor of knowledge required to answer the different questions that might be posed in that domain. Perhaps, e.g., the medium of generic statement lists is not the most effective way to represent the knowledge required to answer questions in the medical domain, and a different medium or representation might be more appropriate for entailment engines.

This work envisions a scenario in which microtheories provide valuable benefits to entailment engines by improving their grounding, efficiency, and accuracy. I observed in my experiments that whether these explicit benefits are realized was unclear for

the scenarios considered. Future work could explore more deeply the relationship between the nature of the domain and the effectiveness of microtheories in improving the performance of entailment engines, and could consider other ways in which microtheories might be used to improve the performance of entailment engines in different domains.

Chapter 7

Conclusions and Future Work

7.1 Summary

I began this thesis by arguing that modern AI systems, particularly large language models (LLMs), could benefit from a reintroduction of core concepts from symbolic AI, such as the systematicity, interpretability, and verifiability provided by systems for automated theorem proving. I then set out my vision for a neuro-symbolic reasoning framework that combines the best of both worlds: the flexible representation and reasoning capabilities of LLMs with the systematic, verifiable reasoning of symbolic search. Core to my approach was the notion of compositional entailment, whereby a hypothesis is deductively supported via premise decompositions grounded in verifiable corpora of natural language text. The entailment engine thus reasons with the skeleton of a hypothesis grounding, backward chaining symbolic solver, but reasons over pure natural language without parsing into brittle formalisms.

I presented a series of systems that embody this vision, the NELLIE (Chapter 3) and TREEWISE (Chapter 5) entailment engines, which perform end-to-end question answering by searching for grounded entailment trees. I showed that these systems can achieve state-of-the-art performance on a version of question answering that requires both answering a question correctly and providing a deductively valid tree showing how the answer follows from knowledge in an external text source. I explored various underlying knowledge sources, from the well-curated WorldTree corpus to large, noisy text corpora like Wikipedia. I showed that, in general, these engine's capabilities scale with the "coverage" of the knowledge store, a similar phenomenon to classical symbolic

reasoners, with the added benefit that my entailment engines can reason over large, readily available corpora of pure NL text.

The neural entailment engine's critical decision process is the generation and verification of decompositional entailment steps that support a hypothesis. I investigated mechanisms for improving the protocol for consistently annotating datasets for this task and for improving the entailment engine's generative and discriminative components. I illustrated this protocol in a newly annotated challenge dataset, RDTE (Recognizing Decompositional Textual Entailment), founded on a consistent and theoretically grounded granular protocol for annotating entailment. We showed that today's LLMs struggle with the RDTE dataset. We also applied the methodology to fill a gap in entailment tree literature by introducing a metric for measuring overall tree quality. We also demonstrated the practical benefits of RDTE through a knowledge distillation pipeline that improves decompositional reasoning (even outperforming the 'teacher' LLM), QA accuracy, and proof quality when employed within the reasoning engine.

Finally, I explored methodologies for automatically curating knowledge buffers deemed "microtheories" that capture generalizable core knowledge for a domain. I showed that these microtheories capture highly reused facts that can be used to improve the performance of entailment engines on new domains.

In summary, this thesis has introduced a neuro-symbolic reasoning framework based on textual entailment and tree search. I have advocated for a version of reasoning–and

consequently, of evaluation–that pursues the complementary goals of strong, precise question answering and of providing transparent, verifiable reasoning traces that reflect the behavior of the system. Building on a substantial foundation of work in NLP on entailment, language generation, text retrieval, and question answering, I have built a series of implementations of the framework that illustrate the feasibility of the entailment tree-based approach that achieves strong performance at both goals using current technologies–namely, neural language models, dense retrieval, and carefully fine-tuned neural classifiers. I have also introduced a methodology for improving the quality of the reasoning traces produced by these systems and for automatically curating knowledge buffers that can improve their performance on new domains.

7.2 Lessons

> [A]n AI system is a complex join of many mechanisms, some new, most familiar. Of necessity, on the edge of the art, systems are messy and inelegant joins–that's the nature of frontiers. It is difficult to extract from these data points the scientific increments that should be added to the cumulation. Thus, AI is case-study science with a vengeance. But if that were not enough of a problem, the payoff structure of AI permits the extraction to be put off, even to be avoided permanently. If a system performs well and breaks new ground–which can often be verified by global output measures and direct qualitative assessment–then it has justified its construction. Global conclusions, packaged as the discursive views of its designers, are often the only increments to be added to the cumulated scientific base.
>
> (Newell, 1984)

Newell, developer of the Logic Theorist, observed that the development of AI systems in the logic programming tradition reflected a long process of gradual refinement and adjustment of a complex system with many moving parts. It is challenging to compile progress on such a system into a series of neat, straightforward theoretical breakthroughs that lead to the system's overall performance. Newell's observation in 1984 matches my experience in the 30 months developing the NELLIE and TREEWISE systems described in the previous chapters: progress on highly modular systems centered on the goal of textual entailment-based hypothesis proving, while grounded in various linguistic theories of AI, linguistics, and logic programming, was a marathon rather than a sprint. Scientific findings that in other settings might take the form of "using approach X instead of Y yields better performance for module Z" for me always required the added qualifier "in the context of the current broader system W at

the current time of investigation T." Changes, e.g., to the retrieval components of the system, might have a beneficial effect on the end QA performance one week but might be rendered ineffectual the next once I fiddled with the rule generation components. It is, therefore, difficult to put into words (to the detriment of my endeavors with peer review) the breakthroughs that led to the systems, which I continue to refine and improve today. I attempted to lay out in sections such as Section 3.5 and Section 5.4.8 the kinds of design choices that consistently yielded empirical improvements to the system's QA performance, though these sections provide only a partial picture of a substantially murkier investigation. Moreover, my observations should be taken in the context of the specific moment(s) in AI research I pursued these projects and the context of building a system designed for complex compositional entailment.

However, I can still draw some general lessons from the development of these systems, which I will now outline.

7.2.1 It is easy to find local maxima

A major lesson learned in this process is that building a neuro-symbolic search engine is itself a search guided by the heuristics of the NLP researcher(s) seeking to put current techniques to practice in a fully functional reasoning system that reflects a broader vision. The search space is vast, and the space of possible improvements to the system is even vaster, from fiddling with various search parameters at various levels of significance (how many candidates to generate? how many to keep? how many facts

to retrieve? what prompt to use? which/how many models to use for filtering? which to use for scoring? etc.). It was very easy to find a local maximum in this space, only to switch to a different configuration days later that render the "research question" of a given experiment moot. A lesson one must learn when developing this kind of system is to recognize when such fiddling is no longer productive and to move on to a different aspect of the framework that might yield more substantial improvements.

7.2.2 The precision-recall tradeoff is a constant factor

The entailment engine can be considered a binary classifier: either a hypothesis is groundable to a corpus, or it is not. This obscures the fact that an entailment search comprises many thousands of sub-decisions, each of which can be considered a binary entailment decision in its own right. As no entailment model is perfectly precise, one finds quickly that if you try enough decompositions, you will fool the model into accepting a hypothesis that is not actually entailed by the corpus. However, if you try too few decompositions or too few facts, you risk missing a valid decomposition that would have led to a correct proof.[1] This is a classic precision-recall tradeoff, and it was a frequent struggle to find the right balance between the two. The configuration described in Section 5.4.8 was a compromise between the two for the purpose of QA, allowing for imprecise results to supersede precise ones only when the latter were not found.

[1]Reducing the budget, in fact, can improve the system's precision to the point of improving QA accuracy at one point. This occurred in one instance of adding NELLIE's template-guided generation, which slowed down the system compared to not using it and thus accidentally improved precision.

7.2.3 Entailment is contextual

My work in Chapter 4 on eliciting consistent human preferences for compositional entailment (Weir et al., 2024b) informs the intuitions that (A) deductive validity is domain-specific and non-monolithic and (B) for an explanation such as an entailment tree meant to support an agent's decision, its quality aligns with the extent to which it would *convince* a human audience of its conclusion to a sufficient degree of confidence (Holloway et al., 2021). However, there are many possible degrees of confidence at which to set this threshold, and there is no one set of entailment rules that is universally valid across all domains. By choosing to use entailment as the underlying principle for logical deduction (as opposed to strict logical deduction), we accept that the validity of a conclusion is not absolute but rather depends on the context in which it is presented. We moreover found discrepancies in what constitutes validity even between the two domains we considered in RDTE (ARC and HotpotQA), necessitating changes to our annotation rubrics to account for these differences. This suggests that practitioners must carefully consider the context in which their entailment engine will be used and adjust systems for discerning validity accordingly.

7.2.4 Generation as a subroutine is inefficient

In the evaluations of NELLIE in Chapter 3, I set the timeout for the engine to be 180 seconds per question option, i.e., up to 12 minutes per question. This is impractical

for a real-world system, and it is clear that the generation component of the system is a bottleneck. The TreeWise engine is even slower on account of the numerous new and revised modules that use LM prompting as opposed to classifiers. Out of necessity (LLM prompting yielded more precise and flexible reasoning over harder decompositions), a backward-chaining neural model-based entailment engine is slow and deliberate. This is a major drawback of the system, though future developments in LLM generation and revisions of the search logic to be more frugal or pragmatic about which search nodes to explore could alleviate this issue.

7.3 Discussion and Future Work

This thesis has explored the role of entailment engines in modern AI systems based on the intuitions that parts of the classical symbolic reasoning-as-proof-search paradigm are useful to reintroduce into modern neural-based systems. I have developed one embodiment of this vision in the form of the entailment engine, a system that reasons over natural language text by searching for grounded entailment trees. However, I hesitate to argue that all AI systems should use entailment engines for reasoning tasks. Rather, I see entailment engines as a tool for achieving certain behavioral properties in AI that are not easily achieved with other currently popular methods, albeit at the cost of speed and efficiency.

7.3.1 The Role of Entailment Engines Moving Forward

Historically, many symbolic AI researchers treated interpretability and verifiability as first-class citizens in their systems; MYCIN, considered the first expert system, was crafted carefully with transparency and verifiability in mind (Buchanan and Shortliffe, 1984). As GOFAI systems evolved, the properties of interpretability, verifiability, and groundedness were a nice side effect of the paradigm that dominated the field at the time in the absence of more inferentially powerful and/or easy-to-implement methods. Yet, as the field of AI has progressed, these properties have been largely abandoned in favor

of more efficient, opaque, and data-driven methods that practitioners have found to be empirically more effective and easier to implement (training an LLM on a large gob of text is arguably less burdensome than carefully curating a granular knowledge base of facts and rules). Had the NLP not abandoned the design requirements of groundedness and verifiability, we likely would not have seen the rise of LLMs and the subsequent explosion of increasingly powerful models that have shocked seasoned researchers with their capabilities and have come to dominate both academic and industrial research today. Yet, as explainability and systematicity are no longer a side-effect of the dominant reasoning paradigm, they have become a "bonus" part of reasoning that many can get away with disregarding. Whether to pour resources and research time into developing systems that are more interpretable, verifiable, and grounded is a calculation that researchers must make based on the requirements of their problem domain and the tradeoffs they are willing to make. I hope that the work in this thesis can provide a proof of concept for explainable systematic reasoning that can inform this decision calculus.

Reintroducing these constraints into modern AI systems without sacrificing the speed and efficiency of just calling an LLM is a difficult task– for the various QA tasks I explored in this thesis, entailment engines are slow and require many calls to the LLM and ultimately rarely exceed the end-task (i.e. QA) performance of a simple, ungrounded LLM-querying baseline. However, there are certain tasks for which the properties of interpretability, verifiability, and groundedness are worth the cost of speed and efficiency. As it stands, the future role that neural entailment

engines might play in AI systems depends on either the specific requirements of a given problem or hereto unexplored improvements to the entailment engine framework that make it more efficient and scalable than it currently is. Below, I will discuss both of these considerations: when one might consider using an entailment engine and what improvements might be made to the entailment engine framework to make it more viable for a wider range of tasks.

7.3.1.1 When Should I Use an Entailment Engine?

Researchers should consider using an entailment engine when their problem at hand has some combination of the following (non-exhaustive) list of properties:

1. The problem requires and/or benefits from combining discrete knowledge from a large store of discrete knowledge or evidence

2. The problem is conducive to offline inference, planning, or long inference-time computations (more on this below)

3. The problem requires a high degree of interpretability to an end user, e.g., a legal or medical professional, a scientist, or a layperson who is interested in understanding the reasoning behind a decision

4. The problem requires and/or benefits from a high degree of directed verifiability and groundedness

5. There exists (or there can be constructed) a corpus of knowledge that is assumed to be correct, complete, and nontrivially relevant to solving the problem

6. The problem can be posed as a series of decompositional entailments, i.e., the researcher can envision an ideal reasoning solution to a typical problem instance as a series of entailment steps that support a hypothesis

7. What denotes valid decompositional entailments can be clearly codified and annotated by humans

8. The problem can be posed as hypothesis proving (or comparing multiple conflicting hypotheses) without jumping through many unnatural hoops to achieve such a recast (e.g., many embodied RL agent tasks are not conducive to this kind of conversion).

In general, the entailment engine framework is not a general-purpose reasoning engine that replaces a chatbot on your phone. Often, the typical user of Siri, Alexa, or Google Assistant wants a quick, accurate answer to a question and is unwilling to sacrifice wait times for the sake of receiving a more verifiable, grounded, and interpretable answer. Similarly, many currently envisioned applications of "AI Assistants," e.g., any used in real time to assist a human in making an immediate decision in the medical or legal fields, require that the system be efficient at inference time to be practically useful.

There are many tasks, however, for which the properties of interpretability and

groundedness are worth the cost of speed and efficiency either because of the importance of avoiding errors or because inference-time speed is not a primary concern and deliberative reasoning is permissible. Some examples of such tasks are described below:

7.3.1.2 Entailment Engines as Verification Modules

One potential use of entailment engines is as a post-hot verification module for state-of-the-art, otherwise ungrounded question-answering systems. Fast LLM-querying methods are more performative than entailment engines at test time, but they are also more prone to hallucinations and errors (Ji et al., 2022) that might be detected using a system that inherently considers whether the answer follows as a direct logical consequence of retrieved information. I envision a system in which a fast LLM-querying system generates a response to a question, and then an entailment engine flags responses that are not supported by the knowledge in the entailment engine's knowledge store. This flagging could occur either during inference time (with the consequent slowdown, though perhaps only for answers for which the LLM is uncertain) or offline during the LLM's post-training phase.

7.3.1.3 Growing and Revising Theories

My work in this thesis has primarily considered static resources. Building on this paradigm, one might consider methods to *dynamically construct* theories for new problem domains, particularly those for which an existing one is not already

pre-curated. In proposing symbolic theorem proving, by separating the theory from the reasoner, McCarthy envisioned a system whose "behaviour will be improvable merely by making statements to it" (McCarthy, 1959): if knowledge about a problem changes or a new kind of problem arises, we can adapt our reasoner by adding new statements to its theory. Accordingly, we can apply a robust, systematic entailment search algorithm such as NELLIE or TREEWISE to a more complex problem if we can automatically develop and verify a theory containing the requisite knowledge. The microtheories project described in Chapter 6 is one example of work in this direction, though the microtheories are evaluated statically and are not updated as new data arrives or as the system performs reasoning tasks (i.e., QA) and receives external feedback. The microtheories project also considered only domains for which there was a priori expected to be a shared body of knowledge in the form of memorizable generic facts that would collectively help in knowledge-intensive tasks that require rote memorization. Many other domains do not have this property, e.g., in grounded task domains, knowledge must be learned from experience. How an entailment engine might learn and revise its theories in such a domain is an open question.

7.3.1.4 AI for Scientific Exploration

This thesis primarily explored methods for proving/disproving hypotheses for which the answer is already known by humanity and the scientific community; what changes are necessary to the artificial reasoning paradigm when considering a new hypothesis

about the world that nobody has ever thought of before? A natural starting point is to consider the ways by which human scientists formulate and then robustly test their new hypotheses about the world: synthesizing new ideas from existing literature using principled inductive reasoning, then carefully designing experiments to test the ideas in a controlled setting.

7.3.2 Short-term Improvements to Entailment Engines

What must be done to make entailment engines more viable for practical use in a wider range of tasks? The primary answer is speed: the current entailment engines are slow and require many calls to the LLM, each of which can take multiple seconds, especially those that require careful, verbose prompts with many input and output tokens (e.g., Figure 4.7). This is a problem for many practical applications of entailment engines, particularly those that require real-time or near-real-time responses. Future work should consider more sophisticated methods that use LLM calls more efficiently, use smaller models, and/or reduce the total number while maintaining the same level of systematicity and precision. This might involve improvements to the search algorithm, e.g., using a more sophisticated candidate selection or branch pruning/consolidation strategy, or improvements to the generative components of the entailment engine. Some of this may come just with new generations of LLMs; I observed during the

development of my entailment engines that as technologies changed and LMs evolved, later generations were more effective at generating decompositions (i.e., a higher fraction of generated candidates would be considered "good"). Another straightforward improvement would be increasing the parallelism of LLM calls (though TreeWise already makes heavy use of APIs like OpenAI, which call multiple copies of a given LLM in parallel, a luxury that is not available in all cases).

Another avenue for improvement might be through the retrieval component; the entailment engines in this thesis rely on a relatively simple implementation of IR using a dense retrieval encoder. While this was sufficient for the tasks explored in this thesis, future work might consider more sophisticated methods for candidate set selection. These methods might make better use of the time budget by filtering out irrelevant or ungroundable subqueries more quickly. I observed for medical QA that the retrieval component benefited from query augmentation strategies in which the LLM generated "thoughts" or justifications for the query that were then used to retrieve documents relevant to secondary entities or concepts necessary to fully support the query– for instance, names of drugs or diseases that were not mentioned in the query but were implicitly necessary to answer the question correctly. Progress in the direction of more sophisticated retrieval strategies might be hampered for many tasks for which there exists no evaluation sets; e.g., when I applied the entailment engine to EntailmentBankQA using Wikipedia as the underlying knowledge store, I had no way of knowing whether Wikipedia actually contained the facts necessary to

answer the questions in the dataset, since the dataset and the corpus were created separately. I thus had no way of knowing, beyond manual examination, whether the engine was failing because of a lack of coverage in the knowledge store or because of a failure in the retrieval or decomposition generation components. Future work might consider methods for automatically evaluating the quality of the retrieval component even when the underlying knowledge store and the task dataset are not constructed in tandem. Until then, the only signal for success is the engine's end-task performance, as is the case for many modules in the entailment engine framework. This was the case for the decompositional entailment recognition component until we constructed the RDTE benchmark, which allowed us to evaluate the precision of the entailment filters and furthered the development of the engine framework; this can serve as an archetype for improving different modules via carefully constructed benchmarks. Other areas for which targeted module evaluation could be useful are decomposition generation (gauging how frequently the generator produces correct decompositions) and forward-chaining inference generator (gauging how frequently the generator produces "useful" inferences from retrieved documents under some usefulness metric).

7.3.2.1 Reconciling Knowledge Conflicts

The entailment engines in this thesis are designed to reason over a single, consistent knowledge store, with the implicit assumption that the knowledge is correct and without conflict, reflecting a single, coherent view of the world. This is, naturally, an

idealization and not reflective of the real world. When reasoning over e.g., the internet, we must consider the possibility of conflicting information due to factors like temporal drift, differing perspectives, or outright misinformation. In a recent example, Google rolled out a generative AI model for answering user queries to their search engine; users quickly found that the model was generating false and harmful information not because the model was faulty, but because it was interfacing with documents on the web that contained false information (Titcomb, 2024). Various works in NLP have begun exploring how to reconcile current techniques in the presence of conflicting information (Xu et al., 2024); how this would be done in the context of entailment engines is open for exploration. Progress on this front would start with identifying a suitable (and realistic) problem domain in which (A) there are substantial knowledge conflicts that might impede reasoning and (B) composing knowledge from multiple sources is frequently necessary to solve the problem.

Bibliography

Kareem Ahmed, Stefano Teso, Kai-Wei Chang, Guy Van den Broeck, and Antonio Vergari (2022). "Semantic probabilistic layers for neuro-symbolic learning". *Advances in Neural Information Processing Systems* 35, pages 29944–29959 (cited on page 38).

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama (2019). "Optuna: A next-generation hyperparameter optimization framework". *Proceedings of SIGKDD* (cited on page 80).

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein (2016). "Neural module networks". *Proceedings of the IEEE conference on computer vision and pattern recognition* (cited on page 35).

Forough Arabshahi, Jennifer Lee, Antoine Bosselut, Yejin Choi, and Tom Mitchell (2021a). "Conversational Multi-Hop Reasoning with Neural Commonsense Knowledge and Symbolic Logic Rules". *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational

Linguistics, pages 7404–7418. DOI: 10.18653/v1/2021.emnlp-main.588 (cited on page 34).

Forough Arabshahi, Jennifer Lee, Mikayla Gawarecki, Kathryn Mazaitis, Amos Azaria, and Tom Mitchell (2021b). "Conversational Neuro-Symbolic Commonsense Reasoning". *Proceedings of AAAI* (cited on page 34).

Akari Asai and Hannaneh Hajishirzi (2020). "Logic-Guided Data Augmentation and Regularization for Consistent Question Answering". *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 5642–5650. DOI: 10.18653/v1/2020.acl-main.499 (cited on page 97).

Samy Badreddine, Artur S. d'Avila Garcez, Luciano Serafini, and Mike Spranger (2020). "Logic Tensor Networks". *Artif. Intell.* 303, page 103649. URL: https://api.semanticscholar.org/CorpusID:229678739 (cited on page 38).

Yoshua Bengio, Gary Marcus, and Vincent Boucher (2019). *AI DEBATE! Yoshua Bengio vs Gary Marcus*. Montreal.AI. URL: https://montrealartificialintelligence.com/aidebate/ (cited on page 28).

Luisa Bentivogli, Ido Dagan, and Bernardo Magnini (2017). "The recognizing textual entailment challenges: Datasets and methodologies". *Handbook of linguistic annotation*, pages 1119–1147 (cited on page 47).

Jonathan Berant, Ido Dagan, and Jacob Goldberger (2011). "Global Learning of Typed Entailment Rules". *Proceedings of the 49th Annual Meeting of the Association*

for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, pages 610–619. URL: https://aclanthology.org/P11-1062 (cited on page 50).

Gregor Betz, Christian Voigt, and Kyle Richardson (2021). "Critical Thinking for Language Models". *Proceedings of the 14th International Conference on Computational Semantics (IWCS)*. Association for Computational Linguistics, pages 63–75. URL: https://aclanthology.org/2021.iwcs-1.7 (cited on pages 30, 37).

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen-tau Yih, and Yejin Choi (2019). "Abductive Commonsense Reasoning". *International Conference on Learning Representations* (cited on pages 30, 45).

Patrick Blackburn and Johannes Bos (2005). *Representation and inference for natural language: A first course in computational semantics* (cited on page 22).

J Anthony Blair (2012). "Relevance, acceptability and sufficiency today". *Groundwork in the Theory of Argumentation: Selected Papers of J. Anthony Blair*, pages 87–100 (cited on pages 107–109).

J. Anthony Blair and Ralph H. Johnson (1987). "The Current State of Informal Logic". *Informal Logic* 9. URL: https://api.semanticscholar.org/CorpusID:56103435 (cited on page 106).

BIBLIOGRAPHY

Paul Blair, RV Guha, and Wanda Pratt (1992). "Microtheories: An ontological engineer's guide". *Austin, TX: MCC Technical Report. CYC-050-92* (cited on pages 160, 164).

Johan Bos (2014). "Recognizing textual entailment and computational semantics". *Computing Meaning: Volume 4*. Springer, pages 89–105 (cited on pages 49, 168, 170).

Johan Bos (2015). "Open-Domain Semantic Parsing with Boxer". *Proceedings of the 20th Nordic Conference of Computational Linguistics (NODALIDA 2015)*. Linköping University Electronic Press, Sweden, pages 301–304. URL: https://aclanthology.org/W15-1841 (cited on page 49).

Johan Bos and Katja Markert (2005). "Recognising Textual Entailment with Logical Inference". *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 628–635. URL: https://aclanthology.org/H05-1079 (cited on pages 22, 49, 168).

Antoine Bosselut, Ronan Le Bras, and Yejin Choi (2021). "Dynamic Neuro-Symbolic Knowledge Graph Construction for Zero-shot Commonsense Question Answering". *AAAI* (cited on pages 34, 58).

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi (2019). "COMET: Commonsense Transformers for Automatic Knowledge Graph Construction". *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for

Computational Linguistics, pages 4762–4779. DOI: 10.18653/v1/P19-1470 (cited on page 34).

Kaj Bostrom, Zayne Sprague, Swarat Chaudhuri, and Greg Durrett (2022). "Natural Language Deduction through Search over Statement Compositions". *Findings of the Association for Computational Linguistics: EMNLP 2022*. Association for Computational Linguistics, pages 4871–4883. DOI: 10.18653/v1/2022.findings-emnlp.358 (cited on pages 51, 58, 102).

Adam Bouyamourn (2023). "Why LLMs Hallucinate, and How to Get (Evidential) Closure: Perceptual, Intensional, and Extensional Learning for Faithful Natural Language Generation". *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 3181–3193. DOI: 10.18653/v1/2023.emnlp-main.192 (cited on page 26).

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning (2015). "A large annotated corpus for learning natural language inference". *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 632–642. DOI: 10.18653/v1/D15-1075 (cited on pages 45, 47, 126).

Gerhard Brewka, Thomas Eiter, and Miroslaw Truszczynski (2011). "Answer set programming at a glance". *Communications of the ACM* 54, pages 92 –103. URL: https://api.semanticscholar.org/CorpusID:17746168 (cited on page 19).

BIBLIOGRAPHY

Dana Brin, Vera Sorin, Akhil Vaid, Ali Soroush, Benjamin S Glicksberg, Alexander W Charney, Girish Nadkarni, and Eyal Klang (2023). "Comparing ChatGPT and GPT-4 performance in USMLE soft skill assessments". *Scientific Reports* 13.1, page 16492 (cited on page 5).

Bruce G Buchanan and Edward H Shortliffe (1984). *Rule based expert systems: the mycin experiments of the stanford heuristic programming project (the Addison-Wesley series in artificial intelligence)*. Addison-Wesley Longman Publishing Co., Inc. (cited on pages 3, 21, 230).

Jifan Chen, Eunsol Choi, and Greg Durrett (2021). "Can NLI Models Verify QA Systems' Predictions?" *Findings of the Association for Computational Linguistics: EMNLP 2021*. Association for Computational Linguistics, pages 3841–3854. DOI: 10.18653/v1/2021.findings-emnlp.324 (cited on page 48).

Tongfei Chen, Zhengping Jiang, Adam Poliak, Keisuke Sakaguchi, and Benjamin Van Durme (2020). "Uncertain Natural Language Inference". *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 8772–8779. DOI: 10.18653/v1/2020.acl-main.774 (cited on pages 42, 43, 45, 126).

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova (2019). "BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions". *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human*

Language Technologies, Volume 1 (Long and Short Papers). Association for Computational Linguistics, pages 2924–2936. DOI: 10.18653/v1/N19-1300 (cited on page 45).

Peter Clark, Niranjan Balasubramanian, Sumithra Bhakthavatsalam, Kevin Humphreys, Jesse Kinkead, Ashish Sabharwal, and Oyvind Tafjord (2014). "Automatic construction of inference-supporting knowledge bases". *4th Workshop on Automated Knowledge Base Construction (AKBC)* (cited on page 22).

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord (2018). "Think you have solved question answering? try arc, the ai2 reasoning challenge". *arXiv:1803.05457* (cited on pages 50, 79, 110).

Peter Clark, Oren Etzioni, Tushar Khot, Daniel Khashabi, Bhavana Mishra, Kyle Richardson, Ashish Sabharwal, Carissa Schoenick, Oyvind Tafjord, Niket Tandon, et al. (2020). "From 'F'to 'A'on the NY regents science exams: An overview of the aristo project". *Ai Magazine* 41.4, pages 39–53 (cited on page 50).

Peter Clark, Phil Harrison, John Thompson, William Murray, Jerry Hobbs, and Christiane Fellbaum (2007). "On the Role of Lexical and World Knowledge in RTE3". *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*. Association for Computational Linguistics, pages 54–59. URL: https://aclanthology.org/W07-1409 (cited on page 42).

Peter Clark, Philip Harrison, and Xuchen Yao (2012). "An Entailment-Based Approach to the QA4MRE Challenge." *CLEF (Online Working Notes/Labs/Workshop)* (cited on page 47).

Peter Clark, Bhavana Dalvi Mishra, and Oyvind Tafjord (2023). "BaRDa: A Belief and Reasoning Dataset that Separates Factual Accuracy and Reasoning Ability". *arXiv preprint arXiv:2312.07527* (cited on pages xxi, 46, 104, 114, 118, 123, 126).

Peter Clark, Oyvind Tafjord, and Kyle Richardson (2021). "Transformers as soft reasoners over language". *Proceedings of IJCAI* (cited on page 37).

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* (cited on page 32).

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov (2018). "XNLI: Evaluating Cross-lingual Sentence Representations". *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 2475–2485. DOI: 10.18653/v1/D18-1269 (cited on page 45).

Robin Cooper, Richard Crouch, Jan van Eijck, Chris Fox, Josef van Genabith, Jan Jaspars, Hans Kamp, Manfred Pinkal, David Milward, Massimo Poesio, Stephen Pulman, Ted Briscoe, Holger Maier, and Karsten Konrad (1996). *Using the Framework*. Technical report. FraCaS deliverable D16, 136 pages, also available

by anonymous ftp from ftp://ftp.cogsci.ed.ac.uk/pub/FRACAS/del16.ps.gz. FraCaS: A Framework for Computational Semantics (cited on page 44).

Ido Dagan, Oren Glickman, and Bernardo Magnini (2005). "The PASCAL Recognising Textual Entailment Challenge". *Machine Learning Challenges Workshop*. URL: https://api.semanticscholar.org/CorpusID:8587959 (cited on pages 40, 41, 44, 103, 126).

Bhavana Dalvi, Peter Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark (2021). "Explaining Answers with Entailment Trees". *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 7358–7370. DOI: 10.18653/v1/2021.emnlp-main.585 (cited on pages 7, 50, 52, 58, 64, 67, 75, 79, 102, 140, 162).

Bhavana Dalvi Mishra, Oyvind Tafjord, and Peter Clark (2022). "Towards Teachable Reasoning Systems: Using a Dynamic Memory of User Feedback for Continual System Improvement". *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 9465–9480. DOI: 10.18653/v1/2022.emnlp-main.644 (cited on pages 51, 216).

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum (2021). "Case-based Reasoning for Natural Language Queries over Knowledge Bases". *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Association for Computational Linguistics, pages 9594–9611. DOI: 10.18653/v1/2021.emnlp-main.755 (cited on page 68).

Luc De Raedt, Angelika Kimmig, and Hannu Toivonen (2007). "Problog: A probabilistic prolog and its application in link discovery." *IJCAI*. Volume 7. Hyderabad, pages 2462–2467 (cited on page 20).

Dorottya Demszky, Kelvin Guu, and Percy Liang (2018). "Transforming question answering datasets into natural language inference datasets". *arXiv:1809.02922* (cited on pages 48, 62, 75, 110, 139).

Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan (2024). "Investigating Data Contamination in Modern Benchmarks for Large Language Models". *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Edited by Kevin Duh, Helena Gomez, and Steven Bethard. Mexico City, Mexico: Association for Computational Linguistics, pages 8706–8719. DOI: 10.18653/v1/2024.naacl-long.482. URL: https://aclanthology.org/2024.naacl-long.482 (cited on page 29).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). "Bert: Pre-training of deep bidirectional transformers for language understanding". *arXiv:1810.04805* (cited on page 25).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding".

Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). Association for Computational Linguistics, pages 4171–4186. DOI: 10.18653/v1/N19-1423 (cited on page 48).

Abhimanyu Dubey et al. (2024). *The Llama 3 Herd of Models*. arXiv: 2407.21783 [cs.AI]. URL: https://arxiv.org/abs/2407.21783 (cited on page 29).

Kevin Ellis, Lucas Morales, Mathias Sablé-Meyer, Armando Solar-Lezama, and Josh Tenenbaum (2018). "Learning libraries of subroutines for neurally–guided bayesian program induction". *Advances in Neural Information Processing Systems* 31 (cited on page 215).

Kevin Ellis, Catherine Wong, Maxwell Nye, Mathias Sablé-Meyer, Lucas Morales, Luke Hewitt, Luc Cary, Armando Solar-Lezama, and Joshua B. Tenenbaum (2021). "DreamCoder: bootstrapping inductive program synthesis with wake-sleep library learning". *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*. PLDI 2021. Virtual, Canada: Association for Computing Machinery, 835–850. ISBN: 9781450383912. DOI: 10.1145/3453483.3454080. URL: https://doi.org/10.1145/3453483.3454080 (cited on page 215).

Christiane Fellbaum (2010). "WordNet". *Theory and applications of ontology: computer applications*. Springer, pages 231–243 (cited on page 49).

Kenneth D Forbus (2010). "Modeling amidst the Microtheories". *Proceedings of the 24th International Workshop on Qualitative Reasoning*. Citeseer, pages 112–115 (cited on page 215).

Noah S Friedland, Paul G Allen, Gavin Matthews, Michael Witbrock, David Baxter, Jon Curtis, Blake Shepard, Pierluigi Miraglia, Jurgen Angele, Steffen Staab, et al. (2004). "Project Halo: Towards a digital Aristotle". *AI magazine* 25.4, pages 29–29 (cited on page 164).

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig (2023). "Pal: Program-aided language models". *International Conference on Machine Learning*. PMLR, pages 10764–10799 (cited on page 31).

Gemini Team et al. (2024). *Gemini: A Family of Highly Capable Multimodal Models*. arXiv: 2312.11805 [cs.CL]. URL: https://arxiv.org/abs/2312.11805 (cited on page 29).

Max Glockner, Ieva Staliunaite, James Thorne, Gisela Vallejo, Andreas Vlachos, and Iryna Gurevych (2021). "AmbiFC: Fact-Checking Ambiguous Claims with Evidence". *Transactions of the Association for Computational Linguistics* 12, pages 1–18. URL: https://api.semanticscholar.org/CorpusID:258987607 (cited on page 126).

Noah D. Goodman, Joshua B. Tenenbaum, and Tobias Gerstenberg (2014). "Concepts in a Probabilistic Language of Thought". URL: https://api.semanticscholar.org/CorpusID:16858487 (cited on page 215).

Bert F. Green, Alice K. Wolf, Carol Chomsky, and Kenneth Laughery (1961). "Baseball: An Automatic Question-Answerer". Los Angeles, California. ISBN: 9781450378727. DOI: 10.1145/1460690.1460714. URL: https://doi.org/10.1145/1460690.1460714 (cited on page 22).

Leo Groarke (2022). "Informal Logic". *The Stanford Encyclopedia of Philosophy*. Edited by Edward N. Zalta and Uri Nodelman. Winter 2022. Metaphysics Research Lab, Stanford University. URL: https://plato.stanford.edu/archives/win2022/entries/logic-informal/ (visited on 2024) (cited on pages 104, 106).

Reto Gubelmann, Ioannis Katis, Christina Marianne Niklaus, and Siegfried Handschuh (2023). "Capturing the Varieties of Natural Language Inference: A Systematic Survey of Existing Datasets and Two Novel Benchmarks". *Journal of Logic, Language and Information*, pages 1–28. URL: https://api.semanticscholar.org/CorpusID:265342081 (cited on page 104).

Han Guo, Ramakanth Pasunuru, and Mohit Bansal (2018a). "Dynamic Multi-Level Multi-Task Learning for Sentence Simplification". *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, pages 462–476. URL: https://aclanthology.org/C18-1039 (cited on page 48).

Han Guo, Ramakanth Pasunuru, and Mohit Bansal (2018b). "Soft Layer-Specific Multi-Task Summarization with Entailment and Question Generation". *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pages 687–697. DOI: 10.18653/v1/P18-1064 (cited on page 48).

Nitish Gupta, Kevin Lin, Dan Roth, Sameer Singh, and Matt Gardner (2020a). "Neural Module Networks for Reasoning over Text". *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=SygWvAVFPr (cited on page 35).

Vivek Gupta, Maitrey Mehta, Pegah Nokhiz, and Vivek Srikumar (2020b). "INFOTABS: Inference on Tables as Semi-structured Data". *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 2309–2324. DOI: 10.18653/v1/2020.acl-main.210 (cited on page 45).

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang (2020). "REALM: Retrieval-augmented language model pre-training". *arXiv:2002.08909* (cited on page 8).

Kyle Hamilton, Aparna Nayak, Bojan Božić, and Luca Longo (2022). "Is neuro-symbolic ai meeting its promises in natural language processing? a structured review". *Semantic Web* Preprint, pages 1–42 (cited on page 28).

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Luke Benson, Lucy Sun, Ekaterina Zubova, Yujie Qiao, Matthew Burtell, David Peng, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, Ansong Ni, Linyong Nan, Jungo Kasai, Tao Yu, Rui Zhang, Shafiq Joty, Alexander R. Fabbri, Wojciech Kryscinski, Xi Victoria Lin, Caiming Xiong, and Dragomir Radev (2022). "FOLIO: Natural Language Reasoning with First-Order Logic". *arXiv preprint arXiv:2209.00840*. URL: https://arxiv.org/abs/2209.00840 (cited on page 32).

Sanda Harabagiu and Andrew Hickl (2006). "Methods for Using Textual Entailment in Open-Domain Question Answering". *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 905–912. DOI: 10.3115/1220175.1220289 (cited on page 47).

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt (2021). "Measuring Massive Multitask Language Understanding". *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=d7KBjmI3GmQ (cited on pages 5, 166).

Carl Hewitt (1969). "PLANNER: A Language for Proving Theorems in Robots". *Proceedings of IJCAI-69, IJCAI, Washington DC* (cited on page 19).

C Michael Holloway, Kimberly S Wasson, and Joby Aviation (2021). "A Primer on Argument Assessment". en. URL: https://ntrs.nasa.gov/citations/20210022807 (cited on page 228).

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi (2019). "The Curious Case of Neural Text Degeneration". *ICLR* (cited on pages 30, 66).

Ruixin Hong, Hongming Zhang, Xintong Yu, and Changshui Zhang (2022). "METGEN: A Module-Based Entailment Tree Generation Framework for Answer Explanation". *Findings of the Association for Computational Linguistics: NAACL 2022*. Association for Computational Linguistics, pages 1887–1905. DOI: 10.18653/v1/2022.findings-naacl.145 (cited on page 51).

Md Mosharaf Hossain, Venelin Kovatchev, Pranoy Dutta, Tiffany Kao, Elizabeth Wei, and Eduardo Blanco (2020). "An Analysis of Natural Language Inference Benchmarks through the Lens of Negation". *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, pages 9106–9118. DOI: 10.18653/v1/2020.emnlp-main.732 (cited on page 44).

Filip Ilievski, Alessandro Oltramari, Kaixin Ma, Bin Zhang, Deborah L McGuinness, and Pedro Szekely (2021). "Dimensions of commonsense knowledge". *Knowledge-Based Systems* 229, page 107347 (cited on page 166).

P Jackson (1986). "Introduction to expert systems". URL: https://www.osti.gov/biblio/5675197 (cited on page 54).

Peter Jansen, Kelly J. Smith, Dan Moreno, and Huitzilin Ortiz (2021). "On the Challenges of Evaluating Compositional Explanations in Multi-Hop Inference: Relevance, Completeness, and Expert Ratings". *Proceedings of the 2021*

Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, pages 7529–7542. DOI: 10.18653/v1/2021.emnlp-main.596 (cited on pages 76, 207, 208).

Peter Jansen, Elizabeth Wainwright, Steven Marmorstein, and Clayton Morrison (2018). "WorldTree: A Corpus of Explanation Graphs for Elementary Science Questions supporting Multi-hop Inference". *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. European Language Resources Association (ELRA). URL: https://aclanthology.org/L18-1433 (cited on page 171).

Theo M. V. Janssen and Thomas Ede Zimmermann (2021). "Montague Semantics". *The Stanford Encyclopedia of Philosophy*. Edited by Edward N. Zalta. Summer 2021. Metaphysics Research Lab, Stanford University (cited on page 40).

Harsh Jhamtani and Peter Clark (2020). "Learning to Explain: Datasets and Models for Identifying Valid Reasoning Chains in Multihop Question-Answering". *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, pages 137–150. DOI: 10.18653/v1/2020.emnlp-main.10 (cited on pages 46, 71, 75).

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung (2022). "Survey of hallucination in natural language generation". *ACM Computing Surveys* (cited on pages 31, 54, 160, 234).

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. (2024). "Mixtral of experts". *arXiv preprint arXiv:2401.04088* (cited on page 186).

Zhijing Jin, Abhinav Lalwani, Tejas Vaidhya, Xiaoyu Shen, Yiwen Ding, Zhiheng Lyu, Mrinmaya Sachan, Rada Mihalcea, and Bernhard Schoelkopf (2022). "Logical Fallacy Detection". *Findings of the Association for Computational Linguistics: EMNLP 2022*. Association for Computational Linguistics, pages 7180–7198. DOI: 10.18653/v1/2022.findings-emnlp.532 (cited on page 127).

Jeff Johnson, Matthijs Douze, and Hervé Jégou (2019). "Billion-scale similarity search with GPUs". *IEEE Transactions on Big Data* (cited on page 65).

Ralph H. Johnson and J. Anthony Blair (1977). "Logical Self-Defense". URL: https://api.semanticscholar.org/CorpusID:60754270 (cited on pages 104, 106, 107).

Karen Sparck Jones, Steve Walker, and Stephen E. Robertson (2000). "A probabilistic model of information retrieval: development and comparative experiments - Part 1". *Inf. Process. Manag.* DOI: 10.1016/S0306-4573(00)00015-7. URL: https://doi.org/10.1016/S0306-4573(00)00015-7 (cited on page 68).

Jaehun Jung, Lianhui Qin, Sean Welleck, Faeze Brahman, Chandra Bhagavatula, Ronan Le Bras, and Yejin Choi (2022). "Maieutic Prompting: Logically Consistent Reasoning with Recursive Explanations". *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational

Linguistics, pages 1266–1279. DOI: 10.18653/v1/2022.emnlp-main.82 (cited on page 54).

Aditya Kalyanpur, Tom Breloff, David Ferrucci, Adam Lally, and John Jantos (2021). "Braid: Weaving Symbolic and Neural Knowledge into Coherent Logical Explanations". *AAAI* (cited on pages 34, 56).

Nora Kassner, Benno Krojer, and Hinrich Schütze (2020). "Are Pretrained Language Models Symbolic Reasoners over Knowledge?" *Proceedings of the 24th Conference on Computational Natural Language Learning*. Association for Computational Linguistics, pages 552–564. DOI: 10.18653/v1/2020.conll-1.45 (cited on page 37).

Henry Kautz (2022). "The third AI summer: AAAI Robert S. Engelmore Memorial Lecture". *AI Magazine* (cited on pages 17, 28, 29).

Henry Kautz and Bart Selman (1999). "Unifying SAT-based and graph-based planning". *IJCAI* (cited on page 23).

Henry A Kautz, Bart Selman, et al. (1992). "Planning as Satisfiability." *ECAI* (cited on page 22).

Mehran Kazemi, Najoung Kim, Deepti Bhatia, Xin Xu, and Deepak Ramachandran (2023). "LAMBADA: Backward Chaining for Automated Reasoning in Natural Language". *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pages 6547–6568. DOI: 10.18653/v1/2023.acl-long.361 (cited on pages 37, 56).

Daniel Khashabi, Erfan Sadeqi Azer, Tushar Khot, Ashish Sabharwal, and Dan Roth (2019). "On the Possibilities and Limitations of Multi-hop Reasoning Under Linguistic Imperfections". *arXiv:1901.02522* (cited on page 87).

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal (2020). "Qasc: A dataset for question answering via sentence composition". *Proceedings of AAAI* (cited on pages 46, 71, 119).

Tushar Khot, Daniel Khashabi, Kyle Richardson, Peter Clark, and Ashish Sabharwal (2021). "Text Modular Networks: Learning to Decompose Tasks in the Language of Existing Models". *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 1264–1279. DOI: 10.18653/v1/2021.naacl-main.99 (cited on page 35).

Tushar Khot, Ashish Sabharwal, and Peter Clark (2017). "Answering Complex Questions Using Open Information Extraction". *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, pages 311–316. DOI: 10.18653/v1/P17-2049 (cited on page 81).

Tushar Khot, Ashish Sabharwal, and Peter Clark (2018). "SciTaiL: A Textual Entailment Dataset from Science Question Answering". *AAAI* (cited on page 75).

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal (2023). "Decomposed Prompting: A Modular Approach

for Solving Complex Tasks". *The Eleventh International Conference on Learning Representations*. URL: https://openreview.net/forum?id=_nGgzQjzaRy (cited on page 35).

Rebecca Knowles (2019). "Interactive and Adaptive Neural Machine Translation". PhD thesis. Johns Hopkins University (cited on page 101).

Souvik Kundu, Tushar Khot, Ashish Sabharwal, and Peter Clark (2019). "Exploiting Explicit Paths for Multi-hop Reading Comprehension". *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 2737–2747. DOI: 10.18653/v1/P19-1263 (cited on page 81).

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov (2019). "Natural Questions: A Benchmark for Question Answering Research". *Transactions of the Association for Computational Linguistics* 7, pages 452–466. DOI: 10.1162/tacl_a_00276 (cited on page 188).

Alice Lai, Yonatan Bisk, and Julia Hockenmaier (2017). "Natural Language Inference from Multiple Premises". *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Asian Federation of

Natural Language Processing, pages 100–109. URL: https://aclanthology.org/I17-1011 (cited on page 46).

Guillaume Lample and François Charton (2020). "Deep Learning For Symbolic Mathematics". *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=S1eZYeHFDS (cited on page 37).

Craig Lemoult (2023). "AI could help doctors make better diagnoses". *NPR* (cited on page 5).

Douglas B. Lenat, R. V. Guha, Karen Pittman, Dexter Pratt, and Mary Shepherd (1990). "Cyc: toward programs with common sense". *Commun. ACM* 33.8, 30–49. ISSN: 0001-0782. DOI: 10.1145/79173.79176. URL: https://doi.org/10.1145/79173.79176 (cited on page 160).

Hector Levesque, Ernest Davis, and Leora Morgenstern (2012). "The winograd schema challenge". *Thirteenth international conference on the principles of knowledge representation and reasoning* (cited on page 44).

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer (2019). "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension". *arXiv:1910.13461* (cited on page 67).

Mike Lewis and Mark Steedman (2013). "Combined Distributional and Logical Semantics". *Transactions of the Association for Computational Linguistics* 1, pages 179–192. DOI: 10.1162/tacl_a_00219 (cited on page 49).

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela (2020). "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks". *ArXiv* abs/2005.11401. URL: https://api.semanticscholar.org/CorpusID:218869575 (cited on pages 9, 27).

Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel (2021). "PAQ: 65 Million Probably-Asked Questions and What You Can Do With Them". *Transactions of the Association for Computational Linguistics* 9, pages 1098–1115. DOI: 10.1162/tacl_a_00415 (cited on page 216).

Zhaoyu Li, Jialiang Sun, Logan Murphy, Qidong Su, Zenan Li, Xian Zhang, Kaiyu Yang, and Xujie Si (2024). "A Survey on Deep Learning for Theorem Proving". *arXiv preprint arXiv:2404.09939* (cited on page 33).

James Lighthill (1973). "Artificial intelligence: A general survey". *Artificial Intelligence: a paper symposium*. Science Research Council London, pages 1–21 (cited on page 24).

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira (2021). "Pyserini: A Python Toolkit for Reproducible Information Retrieval Research with Sparse and Dense Representations". *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*, pages 2356–2362 (cited on page 145).

BIBLIOGRAPHY

Nelson F. Liu, Roy Schwartz, and Noah A. Smith (2019). "Inoculation by Fine-Tuning: A Method for Analyzing Challenge Datasets". *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, pages 2171–2179. DOI: 10.18653/v1/N19-1225 (cited on page 97).

S Lohr (2023). "AI may someday work medical miracles. For now, it helps do paperwork". *New York Times*. URL: https://www.nytimes.com/2023/06/26/technology/ai-health-care-documentation.html?login=smartlock (cited on page 161).

Qing Lyu, Marianna Apidianaki, and Chris Callison-Burch (2024). "Towards Faithful Model Explanation in NLP: A Survey". *Computational Linguistics*. eprint: https://direct.mit.edu/coli/article-pdf/doi/10.1162/coli_a_00511/2362278/coli_a_00511.pdf. URL: https://doi.org/10.1162/coli_a_00511 (cited on page 51).

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch (2023). "Faithful Chain-of-Thought Reasoning". *ACL*. Association for Computational Linguistics. URL: https://aclanthology.org/2023.ijcnlp-main.20 (cited on page 31).

Adrian Ma and Darian Woods (2024). *One of the hottest jobs in AI right now: 'types-question guy'*. Podcast. Accessed September 10, 2024. URL: https://www.npr.org/podcasts/510325/the-indicator-from-planet-money (cited on page 132).

Bill MacCartney (2009). "Natural Language Inference". PhD thesis. Stanford University (cited on page 41).

Bodhisattwa Prasad Majumder, Bhavana Dalvi Mishra, Peter Jansen, Oyvind Tafjord, Niket Tandon, Li Zhang, Burch Callison-Burch, and Peter Clark (2023). "CLIN: A Continually Learning Language Agent for Rapid Task Adaptation and Generalization". *arXiv* (cited on page 216).

Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt (2018). "Deepproblog: Neural probabilistic logic programming". *NIPS* (cited on pages 60, 71).

Christopher D Manning (2006). "Local textual inference: it's hard to circumscribe, but you know it when you see it–and NLP needs it" (cited on pages 40, 42, 104).

Gary Marcus (2020). "The next decade in AI: four steps towards robust artificial intelligence". *arXiv preprint arXiv:2002.06177* (cited on pages 27, 28).

Gary Marcus and Ernest Davis (2019). *Rebooting AI: Building artificial intelligence we can trust*. Vintage (cited on pages 6, 160).

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli (2014). "A SICK cure for the evaluation of compositional distributional semantic models". *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. European Language Resources Association (ELRA), pages 216–223. URL: http://www.lrec-conf.org/proceedings/lrec2014/pdf/363_Paper.pdf (cited on page 44).

John McCarthy (1959). *Programs with common sense* (cited on pages ii, 16, 17, 83, 96, 235).

Tom McCoy, Ellie Pavlick, and Tal Linzen (2019). "Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference". *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 3428–3448. DOI: 10.18653/v1/P19-1334 (cited on page 44).

Kostas S Metaxiotis, Dimitris Askounis, and John Psarras (2002). "Expert systems in production planning and scheduling: A state-of-the-art survey". *Journal of Intelligent Manufacturing* (cited on pages 20, 54).

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal (2018). "Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering". *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 2381–2391. DOI: 10.18653/v1/D18-1260 (cited on page 83).

Pasquale Minervini, Matko Bošnjak, Tim Rocktäschel, Sebastian Riedel, and Edward Grefenstette (2020). "Differentiable reasoning on large knowledge bases and natural language". *Proceedings of AAAI* (cited on page 32).

Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi, Matt Gardner, Bryan Kisiel,

et al. (2018). "Never-ending learning". *Communications of the ACM* 61.5, pages 103–115 (cited on page 160).

Dan Moldovan, Christine Clark, Sanda Harabagiu, and Steve Maiorano (2003). "COGEX: A Logic Prover for Question Answering". *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 166–172. URL: https://aclanthology.org/N03-1022 (cited on page 22).

Richard Montague et al. (1970). "Universal grammar". *1974*, pages 222–46 (cited on page 39).

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen (2016). "A Corpus and Cloze Evaluation for Deeper Understanding of Commonsense Stories". *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 839–849. DOI: 10.18653/v1/N16-1098 (cited on page 45).

Nasrin Mostafazadeh, Aditya Kalyanpur, Lori Moon, David Buchanan, Lauren Berkowitz, Or Biran, and Jennifer Chu-Carroll (2020). "GLUCOSE: GeneraLized and COntextualized Story Explanations". *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for

Computational Linguistics, pages 4569–4586. DOI: 10.18653/v1/2020.emnlp-main.370 (cited on page 34).

Stephen Muggleton (1991). "Inductive logic programming". *New generation computing* (cited on pages 32, 215).

Mark A Musen and Johan Van der Lei (1988). "Of brittleness and bottlenecks: Challenges in the creation of pattern-recognition and expert-system models". *Machine Intelligence and Pattern Recognition* (cited on pages 21, 23, 54).

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig (2018). "Stress Test Evaluation for Natural Language Inference". *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, pages 2340–2353. URL: https://aclanthology.org/C18-1198 (cited on page 44).

Courtney Napoles-Cohen (2019). "Monolingual Sentence Rewriting as Machine Translation: Generation and Evaluation". PhD thesis. Johns Hopkins University (cited on page 101).

Allen Newell (1984). "Foreword". *Rule based expert systems: the mycin experiments of the stanford heuristic programming project*. Edited by Bruce G Buchanan and Edward H Shortliffe. The Addison-Wesley series in artificial intelligence. Addison-Wesley Longman Publishing Co., Inc., pages xi–xvi (cited on page 225).

Allen Newell and Herbert Simon (1956). "The logic theory machine–A complex information processing system". *IRE Transactions on information theory* 2.3, pages 61–79 (cited on page 16).

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng (2016). "MS MARCO: A human generated machine reading comprehension dataset". *CoCo@ NIPs* (cited on page 75).

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela (2020). "Adversarial NLI: A New Benchmark for Natural Language Understanding". *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 4885–4901. DOI: 10.18653/v1/2020.acl-main.441 (cited on page 47).

Christina Niklaus, Matthias Cetto, André Freitas, and Siegfried Handschuh (2018). "A Survey on Open Information Extraction". *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, pages 3866–3878. URL: https://aclanthology.org/C18-1326 (cited on page 23).

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 (2022). "Show Your Work: Scratchpads for Intermediate Computation with Language Models". *Deep Learning for Code*

Workshop. URL: https://openreview.net/forum?id=HBlx2idbkbq (cited on page 30).

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". *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 5153–5176. DOI: 10.18653/v1/2023.emnlp-main.313 (cited on page 31).

OpenAI (2023). "GPT-4 Technical Report". *ArXiv* abs/2303.08774. URL: https://arxiv.org/abs/2303.08774 (cited on pages 5, 105, 166).

Jiefu* Ou, Nathaniel Weir*, Anton* Belyy, Felix Yu, and Benjamin Van Durme (2021). "InFillmore: Frame-Guided Language Generation with Bidirectional Context". *Proceedings of *SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*. Edited by Lun-Wei Ku, Vivi Nastase, and Ivan Vulić. Online: Association for Computational Linguistics, pages 129–142. DOI: 10.18653/v1/2021.starsem-1.12. URL: https://aclanthology.org/2021.starsem-1.12 (cited on page 13).

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. arXiv: 2203.02155 [cs.CL]. URL: https://arxiv.org/abs/2203.02155 (cited on page 29).

Raquel Mochales Palau and Marie-Francine Moens (2009). "Argumentation mining: the detection, classification and structure of arguments in text". *Proceedings of the 12th International Conference on Artificial Intelligence and Law*. ICAIL '09. Barcelona, Spain: Association for Computing Machinery, 98–107. ISBN: 9781605585970. DOI: 10.1145/1568234.1568246. URL: https://doi.org/10.1145/1568234.1568246 (cited on page 127).

Liangming Pan, Alon Albalak, Xinyi Wang, and William Wang (2023). "Logic-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning". *Findings of the Association for Computational Linguistics: EMNLP 2023*. Association for Computational Linguistics, pages 3806–3824. DOI: 10.18653/v1/2023.findings-emnlp.248 (cited on page 31).

Ellie Pavlick (2017). "Compositional Lexical Semantics in Natural Language Inference". PhD thesis. University of Pennsylvania (cited on page 43).

Ellie Pavlick and Tom Kwiatkowski (2019). "Inherent Disagreements in Human Textual Inferences". *Transactions of the Association for Computational Linguistics* 7, pages 677–694. DOI: 10.1162/tacl_a_00293 (cited on pages 43, 126).

Xiangyu Peng, Siyan Li, Sarah Wiegreffe, and Mark Riedl (2022). "Inferring the Reader: Guiding Automated Story Generation with Commonsense Reasoning". *Findings of the Association for Computational Linguistics: EMNLP 2022*. Association for

Computational Linguistics, pages 7008–7029. DOI: 10.18653/v1/2022.findings-emnlp.520 (cited on page 34).

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller (2019). "Language Models as Knowledge Bases?" *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, pages 2463–2473. DOI: 10.18653/v1/D19-1250 (cited on pages 34, 160).

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* (cited on page 48).

Steven T Piantadosi, Joshua B Tenenbaum, and Noah D Goodman (2016). "The logical primitives of thought: Empirical foundations for compositional cognitive models." *Psychological review* 123.4, page 392 (cited on page 215).

Gabriele Picco, Thanh Lam Hoang, Marco Luca Sbodio, and Vanessa Lopez (2021). "Neural Unification for Logic Reasoning over Natural Language". *Findings of the Association for Computational Linguistics: EMNLP 2021*. Association for Computational Linguistics, pages 3939–3950. DOI: 10.18653/v1/2021.findings-emnlp.331 (cited on page 37).

Gabriel Poesia, Kanishk Gandhi, Eric Zelikman, and Noah Goodman (2024). "Certified Deductive Reasoning with Language Models". *Transactions on Machine Learning*

Research. ISSN: 2835-8856. URL: https://openreview.net/forum?id=yXnwrs2Tl6 (cited on page 36).

Adam Poliak (2020). "Revisiting Recognizing Textual Entailment for Evaluating Natural Language Processing Systems". PhD thesis. Johns Hopkins University (cited on pages 47, 101).

Adam Poliak, Max Fleming, Cash Costello, Kenton Murray, Mahsa Yarmohammadi, Shivani Pandya, Darius Irani, Milind Agarwal, Udit Sharma, Shuo Sun, Nicola Ivanov, Lingxi Shang, Kaushik Srinivasan, Seolhwa Lee, Xu Han, Smisha Agarwal, and Jo

textasciitilde ao Sedoc (2020). "Collecting Verified COVID-19 Question Answer Pairs". *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*. Association for Computational Linguistics. DOI: 10.18653/v1/2020.nlpcovid19-2.31 (cited on page 44).

Adam Poliak, Aparajita Haldar, Rachel Rudinger, J. Edward Hu, Ellie Pavlick, Aaron Steven White, and Benjamin Van Durme (2018). "Collecting Diverse Natural Language Inference Problems for Sentence Representation Evaluation". *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 67–81. DOI: 10.18653/v1/D18-1007 (cited on page 44).

Stanislas Polu and Ilya Sutskever (2020). "Generative Language Modeling for Automated Theorem Proving". *ArXiv* (cited on page 33).

BIBLIOGRAPHY

Emil L Post (1943). "Formal reductions of the general combinatorial decision problem". *American journal of mathematics* 65.2, pages 197–215 (cited on page 19).

Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP (2021). Association for Computational Linguistics. URL: https://aclanthology.org/2021.blackboxnlp-1.0 (cited on page 25).

Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-Rong Wen (2024). "Tool Learning with Large Language Models: A Survey". *arXiv preprint arXiv:2405.17935* (cited on page 36).

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu (2020). "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". *JMLR* (cited on pages 25, 48, 67, 75, 77).

Dheeraj Rajagopal, Aman Madaan, Niket Tandon, Yiming Yang, Shrimai Prabhumoye, Abhilasha Ravichander, Peter Clark, and Eduard Hovy (2021). "Curie: An iterative querying approach for reasoning about situations". *arXiv:2104.00814* (cited on page 30).

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang (2016). "SQuAD: 100,000+ Questions for Machine Comprehension of Text". *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 2383–2392. DOI: 10.18653/v1/D16-1264 (cited on page 44).

Nils Reimers and Iryna Gurevych (2019). "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, pages 3982–3992. DOI: 10.18653/v1/D19-1410 (cited on pages 75, 120, 134, 139, 145).

Justus Renkhoff, Ke Feng, Marc Meier-Doernberg, Alvaro Velasquez, and Houbing Herbert Song (2024). "A Survey on Verification and Validation, Testing and Evaluations of Neurosymbolic Artificial Intelligence". *IEEE Transactions on Artificial Intelligence* 5.8, 3765–3779. ISSN: 2691-4581. DOI: 10.1109/tai.2024.3351798. URL: http://dx.doi.org/10.1109/TAI.2024.3351798 (cited on page 28).

Alexandre Riazanov and Andrei Voronkov (2002a). "The design and implementation of VAMPIRE". *AI communications* (cited on pages 22, 49).

Alexandre Riazanov and Andrei Voronkov (2002b). "The design and implementation of VAMPIRE". *AI communications* (cited on page 170).

Danilo Neves Ribeiro, Shen Wang, Xiaofei Ma, Rui Dong, Xiaokai Wei, Henry Zhu, Xinchi Chen, Zhiheng Huang, Peng Xu, Andrew O. Arnold, and Dan Roth (2022). "Entailment Tree Explanations via Iterative Retrieval-Generation Reasoner". *NAACL-HLT* (cited on pages 51, 58, 103).

Danilo Neves Ribeiro, Shen Wang, Xiaofei Ma, Henghui Zhu, Rui Dong, Deguang Kong, Juliette Burger, Anjelica Ramos, zhiheng huang, William Yang Wang,

George Karypis, Bing Xiang, and Dan Roth (2023). "STREET: A MULTI-TASK STRUCTURED REASONING AND EXPLANATION BENCHMARK". *The Eleventh International Conference on Learning Representations*. URL: https://openreview.net/forum?id=1C_kSW1-k0 (cited on pages 51, 103).

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*. Volume 34. 05, pages 8713–8721 (cited on page 44).

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. (1995a). "Okapi at TREC-3". *Nist Special Publication Sp* 109, page 109 (cited on page 145).

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. (1995b). "Okapi at TREC-3". *Nist Special Publication Sp* 109, page 109 (cited on page 173).

Tim Rocktäschel and Sebastian Riedel (2017). "End-to-end differentiable proving". *Advances in neural information processing systems* (cited on page 19).

Ohad Rozen, Shmuel Amar, Vered Shwartz, and Ido Dagan (2021). "Teach the Rules, Provide the Facts: Targeted Relational-knowledge Enhancement for Textual Inference". *Proceedings of *SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*. Association for Computational Linguistics, pages 89–98. DOI: 10.18653/v1/2021.starsem-1.8 (cited on page 37).

Rachel Rudinger (2019). "Decompositional Semantics for Events, Participants, and Scripts in Text". PhD thesis. Johns Hopkins University (cited on page 101).

Rachel Rudinger, Vered Shwartz, Jena D. Hwang, Chandra Bhagavatula, Maxwell Forbes, Ronan Le Bras, Noah A. Smith, and Yejin Choi (2020). "Thinking Like a Skeptic: Defeasible Inference in Natural Language". *Findings of the Association for Computational Linguistics: EMNLP 2020*. Association for Computational Linguistics, pages 4661–4675. DOI: 10.18653/v1/2020.findings-emnlp.418 (cited on pages 30, 31, 45).

Stuart Russell and Peter Norvig (2010). "Artificial intelligence: a modern approach" (cited on pages 16, 60).

Hyun Ryu, Gyeongman Kim, Hyemin S. Lee, and Eunho Yang (2024). *Divide and Translate: Compositional First-Order Logic Translation and Verification for Complex Logical Reasoning*. arXiv: 2410.08047 [cs.CL]. URL: https://arxiv.org/abs/2410.08047 (cited on page 32).

Mohammed Saeed, Naser Ahmadi, Preslav Nakov, and Paolo Papotti (2021). "RuleBERT: Teaching Soft Rules to Pre-Trained Language Models". *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 1460–1476. DOI: 10.18653/v1/2021.emnlp-main.110 (cited on page 37).

Swarnadeep Saha, Sayan Ghosh, Shashank Srivastava, and Mohit Bansal (2020). "PRover: Proof Generation for Interpretable Reasoning over Rules". *Proceedings*

of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, pages 122–136. DOI: 10.18653/v1/2020.emnlp-main.9 (cited on page 37).

Kate Sanders, Nathaniel Weir, and Benjamin Van Durme (2024a). "TV-TREES: Multimodal Entailment Trees for Neuro-Symbolic Video Reasoning". *arXiv preprint arXiv:2402.19467* (cited on page 14).

Kate Sanders, Nathaniel Weir, and Benjamin Van Durme (2024b). "TV-TREES: Multimodal Entailment Trees for Neuro-Symbolic Video Reasoning". *Proceedings of EMNLP 2024* (cited on page 99).

Roger C Schank (1983). *Dynamic memory: A theory of reminding and learning in computers and people* (cited on page 68).

Lenhart Schubert (2002). "Can we derive general world knowledge from texts?" *Proceedings of the Second International Conference on Human Language Technology Research*. HLT '02. San Diego, California: Morgan Kaufmann Publishers Inc., 94–97 (cited on page 160).

Lenhart Schubert (2015). "Semantic Representation". *Proceedings of AAAI* 29.1. DOI: 10.1609/aaai.v29i1.9759. URL: https://ojs.aaai.org/index.php/AAAI/article/view/9759 (cited on page 23).

Lenhart K Schubert, Benjamin David Van Durme, and Marzieh Bazrafshan (2010). "Entailment inference in a natural logic-like general reasoner". *2010 AAAI Fall Symposium Series* (cited on pages 24, 49, 168).

BIBLIOGRAPHY

A. Carlisle Scott, William J. Clancey, Randall Davis, and Edward H. Shortliffe (1977). "Explanation Capabilities of Production-Based Consultation Systems". *American Journal of Computational Linguistics*. Microfiche 62, pages 1–50. URL: https://aclanthology.org/J77-1006 (cited on pages 3, 21).

Maria I Sessa (2002). "Approximate reasoning by similarity-based SLD resolution". *Theoretical computer science* (cited on page 58).

Edward H. Shortliffe and Bruce G. Buchanan (1975). "A model of inexact reasoning in medicine". *Mathematical Biosciences* 23.3, pages 351–379. ISSN: 0025-5564. DOI: https://doi.org/10.1016/0025-5564(75)90047-4. URL: https://www.sciencedirect.com/science/article/pii/0025556475900474 (cited on page 2).

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, L. Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Vedavyas Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy P. Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis (2016). "Mastering the game of Go with deep neural networks and tree search". *Nature* 529, pages 484–489. URL: https://api.semanticscholar.org/CorpusID:515925 (cited on page 33).

Paul Smolensky, Moontae Lee, Xiaodong He, Wen-tau Yih, Jianfeng Gao, and Li Deng (2016). "Basic reasoning with tensor product representations". *arXiv preprint arXiv:1601.02745* (cited on page 38).

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng (2013). "Reasoning with neural tensor networks for knowledge base completion". *Advances in neural information processing systems* 26 (cited on page 38).

John F Sowa (1993). *Building large knowledge-based systems: Representation and inference in the Cyc Project: DB Lenat and RV Guha* (cited on page 165).

John F Sowa (2006). "The challenge of knowledge soup". *Research trends in science, technology and mathematics education*, pages 55–90 (cited on pages 23, 165).

Christian Stab and Iryna Gurevych (2017). "Recognizing Insufficiently Supported Arguments in Argumentative Essays". *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, pages 980–990. URL: https://aclanthology.org/E17-1092 (cited on page 127).

Roni Stern, Tamar Kulberis, Ariel Felner, and Robert Holte (2010). "Using lookaheads with optimal best-first search". *Proceedings of AAAI* (cited on page 71).

Justin Sulik, Jeroen van Paridon, and Gary Lupyan (2021). "Explanations in the wild". *Cognition* 237. URL: https://api.semanticscholar.org/CorpusID:236620904 (cited on pages 42, 104).

Oyvind Tafjord and Peter Clark (2021). "General-purpose question-answering with macaw". *arXiv preprint arXiv:2109.02593* (cited on page 94).

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark (2021). "ProofWriter: Generating Implications, Proofs, and Abductive Statements over Natural Language". *Findings*

of the Association for Computational Linguistics: ACL-IJCNLP 2021. Association for Computational Linguistics, pages 3621–3634. DOI: 10.18653/v1/2021.findings-acl.317 (cited on page 37).

Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark (2022). "Entailer: Answering Questions with Faithful and Truthful Chains of Reasoning". *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 2078–2093. DOI: 10.18653/v1/2022.emnlp-main.134 (cited on pages 51, 54, 58, 63, 71, 75, 77, 80, 81, 88, 89, 91, 92, 96, 101, 104, 108, 109, 114, 119, 162).

Chenhao Tan (2022). "On the Diversity and Limits of Human Explanations". *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 2173–2188. DOI: 10.18653/v1/2022.naacl-main.158 (cited on pages 42, 51, 104, 127).

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick (2019). "What do you learn from context? Probing for sentence structure in contextualized word representations". *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=SJzSgnRcKX (cited on page 25).

Ramya Keerthy Thatikonda, Jiuzhou Han, Wray Buntine, and Ehsan Shareghi (2024). *Strategies for Improving NL-to-FOL Translation with LLMs: Data Generation, Incremental Fine-Tuning, and Verification*. arXiv: 2409.16461 [cs.CL]. URL: https://arxiv.org/abs/2409.16461 (cited on page 32).

Mokanarangan Thayaparan, Marco Valentino, Deborah Ferreira, Julia Rozanova, and André Freitas (2022). "Diff-Explainer: Differentiable Convex Optimization for Explainable Multi-hop Inference". *Transactions of the Association for Computational Linguistics* 10, pages 1103–1119. DOI: 10.1162/tacl_a_00508 (cited on page 81).

Mokanarangan Thayaparan, Marco Valentino, and André Freitas (2021). "Explainable Inference Over Grounding-Abstract Chains for Science Questions". *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Association for Computational Linguistics, pages 1–12. DOI: 10.18653/v1/2021.findings-acl.1 (cited on pages 56, 81).

James Titcomb (2024). "Google curbs AI search tool after it told people to eat rocks". en-GB. *The Telegraph*. ISSN: 0307-1235. URL: https://www.telegraph.co.uk/business/2024/05/31/google-curbs-ai-search-tool/ (visited on 2024) (cited on page 239).

Aaron Traylor, Roman Feiman, and Ellie Pavlick (2021). "AND does not mean OR: Using Formal Languages to Study Language Models' Representations". *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and*

the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers). Association for Computational Linguistics, pages 158–167. DOI: 10.18653/v1/2021.acl-short.21 (cited on page 37).

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal (2023). "Interleaving Retrieval with Chain-of-Thought Reasoning for Knowledge-Intensive Multi-Step Questions". *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, pages 10014–10037. DOI: 10.18653/v1/2023.acl-long.557 (cited on page 36).

Harsh Trivedi, Heeyoung Kwon, Tushar Khot, Ashish Sabharwal, and Niranjan Balasubramanian (2019). "Repurposing Entailment for Multi-Hop Question Answering Tasks". *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, pages 2948–2958. DOI: 10.18653/v1/N19-1302 (cited on page 47).

Nathaniel Weir, João Sedoc, and Benjamin Van Durme (2020). "COD3S: Diverse Generation with Discrete Semantic Signatures". *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Edited by Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu. Online: Association for Computational Linguistics, pages 5199–5211. DOI: 10.18653/v1/2020.emnlp-main.421. URL: https://aclanthology.org/2020.emnlp-main.421 (cited on page 13).

BIBLIOGRAPHY

Nathaniel Weir, Ryan Thomas, Randolph D'Amore, Kellie Hill, Benjamin Van Durme, and Harsh Jhamtani (2024). "Ontologically faithful generation of non-player character dialogues". *Proceedings of EMNLP* (cited on page 12).

Nathaniel Weir, Xingdi Yuan, Marc-Alexandre Côté, Matthew Hausknecht, Romain Laroche, Ida Momennejad, Harm Van Seijen, and Benjamin Van Durme (2023). "One-Shot Learning from a Demonstration with Hierarchical Latent Language". *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*. AAMAS '23. London, United Kingdom: International Foundation for Autonomous Agents and Multiagent Systems, 2388–2390. ISBN: 9781450394321 (cited on page 14).

Marco Valentino, Ian Pratt-Hartmann, and André Freitas (2021). "Do Natural Language Explanations Represent Valid Logical Arguments? Verifying Entailment in Explainable NLI Gold Standards". *Proceedings of the 14th International Conference on Computational Semantics (IWCS)*. Association for Computational Linguistics, pages 76–86. URL: https://aclanthology.org/2021.iwcs-1.8 (cited on page 127).

Marco Valentino, Mokanarangan Thayaparan, and André Freitas (2022). "Case-Based Abductive Natural Language Inference". *Proceedings of the 29th International Conference on Computational Linguistics*. International Committee on Computational Linguistics, pages 1556–1568. URL: https://aclanthology.org/2022.coling-1.134 (cited on page 171).

Benjamin Van Durme (2009). "Extracting implicit knowledge from text". PhD thesis. University of Rochester (cited on page 23).

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R. Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, Shuning Jin, Berlin Chen, Benjamin Van Durme, Edouard Grave, Ellie Pavlick, and Samuel R. Bowman (2019). "Can You Tell Me How to Get Past Sesame Street? Sentence-Level Pretraining Beyond Language Modeling". *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 4465–4476. DOI: 10.18653/v1/P19-1439 (cited on page 48).

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman (2018). "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding". *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Association for Computational Linguistics, pages 353–355. DOI: 10.18653/v1/W18-5446 (cited on pages 25, 44).

Ben Wang and Aran Komatsuzaki (2021a). *GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model* (cited on page 88).

Ben Wang and Aran Komatsuzaki (2021b). *GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model*. https://github.com/kingoflolz/mesh-transformer-jax (cited on page 90).

Zhong-Ling Wang, Po-Hsien Huang, Wen-Yau Hsu, and Hen-Hsen Huang (2023). "Self-Adapted Utterance Selection for Suicidal Ideation Detection in Lifeline Conversations". *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 1436–1446. DOI: 10.18653/v1/2023.eacl-main.105 (cited on page 35).

Leon Weber, Pasquale Minervini, Jannes Münchmeyer, Ulf Leser, and Tim Rocktäschel (2019). "NLProlog: Reasoning with Weak Unification for Question Answering in Natural Language". *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 6151–6161. DOI: 10.18653/v1/P19-1618 (cited on pages 32, 33, 58, 61, 71, 170).

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, pages 24824–24837 (cited on pages 30, 54, 173).

Nathaniel Weir, Peter Clark, and Benjamin Van Durme (2024a). "NELLIE: A Neuro-Symbolic Inference Engine for Grounded, Compositional, and Explainable Reasoning". *Proceedings of IJCAI 2024*. URL: https://api.semanticscholar.org/CorpusID:258762900 (cited on pages 11, 101, 103, 119, 131, 133, 162).

Nathaniel Weir, Adam Poliak, and Benjamin Van Durme (2020). "Probing Neural Language Models for Human Tacit Assumptions". *Cognitive Science Society*. URL: https://cognitivesciencesociety.org/cogsci20/papers/0070/0070.pdf (cited on page 166).

Nathaniel Weir, Kate Sanders, Orion Weller, Shreya Sharma, Dongwei Jiang, Zhengping Jiang, Bhavana Dalvi Mishra, Oyvind Tafjord, Peter Jansen, Peter Clark, et al. (2024b). "Enhancing systematic decompositional natural language inference using informal logic". *Proceedings of EMNLP 2024* (cited on pages 12, 186, 228).

Margaret Welbank (1983). *A review of knowledge acquisition techniques for expert systems* (cited on page 20).

Sean Welleck, Jiacheng Liu, Ximing Lu, Hannaneh Hajishirzi, and Yejin Choi (2022). "NaturalProver: Grounded Mathematical Proof Generation with Language Models". *arXiv:2205.12910* (cited on page 33).

Orion Weller, Aleem Khan, Nathaniel Weir, Dawn Lawrie, and Benjamin Van Durme (2024a). "Defending Against Disinformation Attacks in Open-Domain Question Answering". *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*. Edited by Yvette Graham and Matthew Purver. St. Julian's, Malta: Association for Computational Linguistics, pages 402–417. URL: https://aclanthology.org/2024.eacl-short.35 (cited on page 12).

Orion Weller, Marc Marone, Nathaniel Weir, Dawn Lawrie, Daniel Khashabi, and Benjamin Van Durme (2024b). ""According to . . . ": Prompting Language Models Improves Quoting from Pre-Training Data". *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*. Edited by Yvette Graham and Matthew Purver. St. Julian's, Malta: Association for Computational Linguistics, pages 2288–2301. URL: https://aclanthology.org/2024.eacl-long.140 (cited on page 12).

Aaron Steven White, Pushpendre Rastogi, Kevin Duh, and Benjamin Van Durme (2017). "Inference is Everything: Recasting Semantic Resources into a Unified Evaluation Framework". *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Asian Federation of Natural Language Processing, pages 996–1005. URL: https://aclanthology.org/I17-1100 (cited on page 44).

Sarah Wiegreffe and Ana Marasovic (2021). "Teach Me to Explain: A Review of Datasets for Explainable Natural Language Processing". *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*. URL: https://openreview.net/forum?id=ogNcxJn32BZ (cited on pages 51, 127).

Jan Wielemaker, Tom Schrijvers, Markus Triska, and Torbjörn Lager (2012). "Swi-prolog". *Theory and Practice of Logic Programming* 12.1-2, pages 67–96 (cited on page 137).

BIBLIOGRAPHY

Adina Williams, Nikita Nangia, and Samuel Bowman (2018). "A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference". *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, pages 1112–1122. DOI: 10.18653/v1/N18-1101 (cited on pages 47, 126).

Lionel Wong, Gabriel Grand, Alexander K. Lew, Noah D. Goodman, Vikash K. Mansinghka, Jacob Andreas, and Joshua B. Tenenbaum (2023). *From Word Models to World Models: Translating from Natural Language to the Probabilistic Language of Thought*. arXiv: 2306.12672 [cs.CL] (cited on page 31).

Kevin Wu, Eric Wu, Ally Cassasola, Angela Zhang, Kevin Wei, Teresa Nguyen, Sith Riantawan, Patricia Shi Riantawan, Daniel E Ho, and James Zou (2024). "How well do LLMs cite relevant medical references? An evaluation framework and analyses". *arXiv preprint arXiv:2402.02008* (cited on page 5).

Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, Xu Zhao, Min-Yen Kan, Junxian He, and Qizhe Xie (2023). "Self-Evaluation Guided Beam Search for Reasoning". *Thirty-seventh Conference on Neural Information Processing Systems*. URL: https://openreview.net/forum?id=Bw82hwg5Q3 (cited on page 36).

Zhengnan Xie, Sebastian Thiem, Jaycie Martin, Elizabeth Wainwright, Steven Marmorstein, and Peter Jansen (2020). "WorldTree V2: A Corpus of Science-Domain Structured Explanations and Inference Patterns supporting Multi-Hop

Inference". English. *Proceedings of the Twelfth Language Resources and Evaluation Conference*. European Language Resources Association, pages 5456–5473. ISBN: 979-10-95546-34-4. URL: https://aclanthology.org/2020.lrec-1.671 (cited on pages 62, 75, 79, 131, 147, 171).

Rongwu Xu, Zehan Qi, Zhijiang Guo, Cunxiang Wang, Hongru Wang, Yue Zhang, and Wei Xu (2024). *Knowledge Conflicts for LLMs: A Survey*. arXiv: 2403.08319 [cs.CL]. URL: https://arxiv.org/abs/2403.08319 (cited on page 239).

Kaiyu Yang and Jia Deng (2021). "Learning Symbolic Rules for Reasoning in Quasi-Natural Language". *arXiv:2111.12038* (cited on pages 32, 215).

Kaiyu Yang, Jia Deng, and Danqi Chen (2022). "Generating Natural Language Proofs with Verifier-Guided Search". *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 89–105. DOI: 10.18653/v1/2022.emnlp-main.7 (cited on pages 51, 103).

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning (2018). "HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering". *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 2369–2380. DOI: 10.18653/v1/D18-1259 (cited on page 188).

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan (2023a). "Tree of Thoughts: Deliberate Problem Solving

with Large Language Models". *Thirty-seventh Conference on Neural Information Processing Systems*. URL: https://openreview.net/forum?id=5Xc1ecxO1h (cited on page 36).

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao (2023b). "ReAct: Synergizing Reasoning and Acting in Language Models". *The Eleventh International Conference on Learning Representations*. URL: https://openreview.net/forum?id=WE_vluYUL-X (cited on page 36).

Xi Ye, Qiaochu Chen, Isil Dillig, and Greg Durrett (2023). "SatLM: Satisfiability-Aided Language Models Using Declarative Prompting". *Thirty-seventh Conference on Neural Information Processing Systems*. URL: https://openreview.net/forum?id=TqW5PL1Poi (cited on page 31).

Wenpeng Yin, Dragomir Radev, and Caiming Xiong (2021). "DocNLI: A Large-scale Dataset for Document-level Natural Language Inference". *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Association for Computational Linguistics, pages 4913–4922. DOI: 10.18653/v1/2021.findings-acl.435 (cited on page 45).

John M. Zelle and Raymond J. Mooney (1996). "Learning to Parse Database Queries Using Inductive Logic Programming". *Proceedings of AAAI*. Portland, Oregon. ISBN: 026251091X (cited on page 22).

Sheng Zhang, Rachel Rudinger, Kevin Duh, and Benjamin Van Durme (2017). "Ordinal Common-sense Inference". *Transactions of the Association for Computational*

Linguistics 5, pages 379–395. DOI: 10.1162/tacl_a_00068 (cited on pages 42, 45, 108, 126).

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica (2023). "Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena". *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. URL: https://openreview.net/forum?id=uccHPGDlao (cited on page 166).

Shaowen Zhou, Bowen Yu, Aixin Sun, Cheng Long, Jingyang Li, and Jian Sun (2022). "A Survey on Neural Open Information Extraction: Current Status and Future Directions". *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*. Edited by Lud De Raedt. Survey Track. International Joint Conferences on Artificial Intelligence Organization, pages 5694–5701. DOI: 10.24963/ijcai.2022/793. URL: https://doi.org/10.24963/ijcai.2022/793 (cited on page 31).

Vita

Nathaniel Weir was born and raised in Connecticut. He graduated *magna cum laude* with an Sc.B. in Applied Mathematics and Computer Science from Brown University in 2019. His honors thesis focused on neural semantic parsing and generalization. He joined the Center for Language and Speech Processing at Johns Hopkins University in 2019 and received an M.S.E. in Computer Science from Johns Hopkins in 2021. Nathaniel is the recipient of a National Science Foundation Graduate Research Fellowship. Nathaniel's research focuses on natural language inference, question answering, neuro-symbolic reasoning, explainability, and knowledge grounding.