## Learning Deductive Reasoning from Synthetic Corpus based on Formal Logic

Terufumi Morishita 1 Gaku Morio 1 Atsuki Yamaguchi 1 Yasuhiro Sogawa 1

## **Abstract**

We study a synthetic corpus based approach for language models (LMs) to acquire logical deductive reasoning ability. The previous studies generated deduction examples using specific sets of deduction rules. However, these rules were limited or otherwise arbitrary, limiting the generalizability of acquired reasoning ability. We rethink this and adopt a well-grounded set of deduction rules based on formal logic theory, which can derive any other deduction rules when combined in a multistep way. Then, using the proposed corpora, which we name FLD (Formal Logic Deduction), we first evaluate and analyze the logical reasoning ability of the latest LLMs. Even GPT-4 can solve only half of the problems, suggesting that pure logical reasoning isolated from knowledge is still challenging for the LLMs, and additional training specialized in logical reasoning is indeed essential. We next empirically verify that LMs trained on FLD corpora acquire more generalizable reasoning ability. Furthermore, we identify the aspects of reasoning ability on which deduction corpora can enhance LMs and those on which they cannot, and discuss future directions on each aspect. The released corpora serve both as learning resources and as challenging benchmarks.

### 1. Introduction

Building a machine that reasons with abundant knowledge has been the Holy Grail since the early era of artificial intelligence (McCarthy, 1959). Recent language models (LMs) have taken a step toward this goal, as they demonstrated extensive factual and commonsense knowledge obtained through large-scale pre-training. Even so, LMs still struggle with logical reasoning (Askell, 2020; Rae et al., 2021; Razeghi et al., 2022; Liu et al., 2023; Turpin et al., 2023;

Proceedings of the  $40^{th}$  International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

Lanham et al., 2023; Wu et al., 2023; Hodel & West, 2023; Dziri et al., 2023; Dasgupta et al., 2023), as it demands rigid manipulation of logical rules rather than merely referring to the existing knowledge.

LMs have acquired their knowledge from a lot of high-quality examples in human-written texts (Devlin et al., 2019). Conversely, their poor logical reasoning ability suggests the lack of high-quality examples of logical reasoning in the texts. This is not a surprise given that humans usually think reflexively rather than logically step by step (Kahneman, 2011). The consideration here suggests a straightforward strategy to endow LMs with logical reasoning ability: create corpora that include many examples of valid logical reasoning, and train LMs on them.

One such corpus is the recently proposed RuleTaker (Clark et al., 2021). RuleTaker is composed of synthetically generated examples of multistep deductive reasoning. A deduction example requires an LM to generate logical steps to (disprove a given hypothesis based on a given set of facts. Each logical step must follow deduction rules of the implication kind, such as  $\forall x F(x) \to G(x), F(a) \vdash G(a)$  (here, ⊢ means "derives"). The facts are randomly constructed except the logical structure (i.e., they have no semantics), therefore referring to existing knowledge never helps solve the task. Artificial Argument Corpus (AACorpus) (Betz et al., 2021) is another synthetic corpus composed of singlestep deduction examples. AACorpus adopted a set of handselected deduction rules useful for critical thinking, such as contraposition  $\mathcal{F} \to \mathcal{G} \vdash \neg \mathcal{G} \to \neg \mathcal{F}$  (¬ is negation). All these corpora teach deductive reasoning, one of the most universally used logical reasoning.

However, it is still an open question whether this research direction will genuinely lead to the improvement of deductive reasoning ability. First, the deduction rules used in the previous corpora were limited or otherwise arbitrary. This can limit the generalizability of the acquired deductive reasoning ability since complex real-world reasoning can require various deduction rules. Second, it has not yet been studied what aspect of deductive reasoning ability deduction corpora can enhance LMs. Such aspects include the ability to solve many-step reasoning, and an understanding of diverse linguistic expressions of logical statements. This investigation is essential to discuss the future directions.

<sup>&</sup>lt;sup>1</sup>Hitachi, Ltd. Research and Development Group, Kokubunji, Tokyo, Japan. Correspondence to: Terufumi Morishita <terufumi.morishita.wp@hitachi.com>.

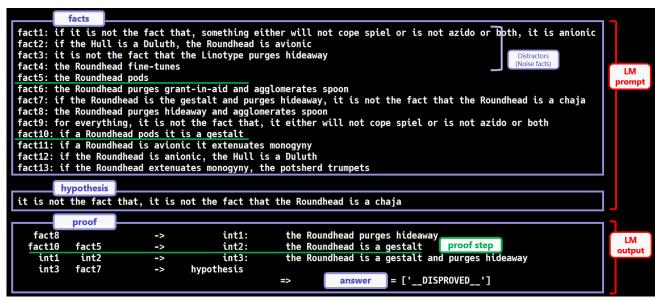


Figure 1: A deduction example generated by **FLD**. Given a set of facts and a hypothesis, an LM is required to generate proof steps to (dis-)prove the hypothesis and an answer. The proof is constructed from a theoretically well-grounded set of deduction rules (i.e., the axioms of first-order predicate logic). Note that the facts are randomly constructed except logical structures (i.e., they have no semantics) so that referring to existing knowledge never helps solve the task.

This paper aims to answer these questions. First, we rethink the choice of deduction rules by leveraging the formal logic theory (Section 2). According to formal logic, there are infinite valid deduction rules, including but not limited to the ones used in the previous corpora. However, among them, there is a set of atomic deduction rules called *the axioms*, and any other valid deduction rules can be derived by multistep deductions constructed from the axioms (*completeness*). Therefore, *the axioms are the most generalizable to various deduction rules*. As the sets of deduction rules used in the previous corpora lack this property, we propose a new framework named **FLD** (Formal Logic **D**eduction), which generates deduction examples constructed from the axioms.

Then, we first investigate how well current (L)LMs perform logical reasoning (Section 5). We find that even the most powerful LLM, GPT-4 (OpenAI, 2023), can solve only half of the problems. More fine-grained analyses reveal several phenomena, notably that the LLMs do not reason faithfully following the "reasoning steps" they themselves generate. Overall, the results here suggest that pure logical reasoning isolated from knowledge is still challenging for latest LLMs, and additional training specialized in logical reasoning should be essential.

Next, we show that the training on **FLD** is effective (Section 6). We trained LMs on **FLD** and measured their performance on two types of benchmarks: one is deduction corpora themselves and the other is human-authored EntailmentBank (EB) (Dalvi et al., 2021), which requires more complex real-world reasoning. The resulst are promising as the LMs outperformed baselines on both benchmarks.

Finally, we identify the aspects of deductive reasoning ability on which deduction corpora are beneficial (Section 7). We analyzed each aspect separately by using a comprehensive set of "ablation corpora", where one corpus emphasizes a specific aspect different from those emphasized by the other corpora. The results suggest that deduction corpora are beneficial in many aspects, but they alone are not enough for some aspects. Finally, on the basis of the results, we discuss the future directions for each aspect (Section 8).

We summarize our contributions as follows:

- We propose<sup>1</sup> a deduction corpus generation framework FLD (Section 3). FLD is the first to leverage formal logic theory, adopting a well-grounded set of deduction rules that generalizes the best to other deduction rules.
- We evaluate and analyze the logical reasoning ability
  of latest LLMs (Section 5). Even GPT-4 can solve
  only half of the problems, suggesting that pure logical
  reasoning isolated from knowledge is still challenging
  for LLMs.
- We empirically verify that LMs trained on FLD corpora acquire more generalizable deductive reasoning ability than the baselines without such training (Section 6).
- We analyze each aspect of deductive reasoning and provide the future directions for applying deduction corpora or other approaches for them (Sections 7 and 8).
- We release<sup>1</sup> the **FLD** corpora, which serve both as learning resources and as challenging benchmarks.

Available at: https://github.com/hitachi-nlp/FLD

## 2. Preliminaries: Formal Logic

Let us consider the following single-step deductive reasoning:

The Earth revolves around the sun, around the sun the Earth has seasons.

The Earth has seasons. (1)

This deduction step derives the conclusion, written under the bar, from the two premises. Next, consider another step:

The Earth revolves around the sun, around the sun the Earth does not have seasons.

The Earth does not have seasons.

In this step, one of the premises (i.e., "If the Earth revolves around the sun, the Earth does not have seasons") is false. However, if the premise had been true, we can still derive the conclusion. Thus, in formal logic, this step is still valid the same as (1). We can abstract (1) and (2) into a deduction rule as:  $\mathcal{F} \longrightarrow \mathcal{F} \longrightarrow \mathcal{G}$  modus papers.

$$\frac{\mathcal{F} \qquad \mathcal{F} \to \mathcal{G}}{\mathcal{G}} \text{ modus ponens} \tag{3}$$

The deduction rule of this form is called *modus ponens*.

While modus ponens is the most intuitive deduction rule, many others exist. For example, a famous syllogism is:

$$\frac{(\mathcal{F} \to \mathcal{G}) \land (\mathcal{G} \to \mathcal{H})}{\mathcal{F} \to \mathcal{H}} \text{ syllogism} \tag{4}$$

The other example below defines the meaning of  $\wedge$  formally:

$$\frac{(\mathcal{F} \wedge \mathcal{G})}{\mathcal{F}} \quad \frac{(\mathcal{F} \wedge \mathcal{G})}{\mathcal{G}} \quad \wedge \text{-elimination} \tag{5}$$

Of course, we can consider invalid  $^2$  rules, in the sense that the conclusion is not logically deducible from the premises, such as:  $\mathcal{F} = (\mathcal{F} \setminus \mathcal{C})$ 

$$\frac{\mathcal{F} \qquad (\mathcal{F} \vee \mathcal{G})}{\mathcal{G}} \tag{6}$$

Now, from these examples, we obtain some important points of deductive reasoning. First, deductive reasoning can be defined as a form of thought in which a conclusion is derived from a set of premises following specific deduction rules, such as the ones in (1) to (6). Such deduction rules are called *arguments* in formal logic theory. Second, whether a deduction rule is valid or not does not depend on contents of symbols but only on the superficial form of the symbolic sequence composed of the premises to the conclusion. For example, as stated above, (3) is valid regardless of the actual content of  $\mathcal{G}$ , such as  $\mathcal{G}$ ="(...), the Earth has seasons." in (1) and  $\mathcal{G}=$ "(...), the Earth does not have seasons." in (2). This enables us to regard all deduction rules simply as symbolic rules such as (3) to (6). Third and as one conclusion of the second point, the symbols such as  $\mathcal{F}$  and  $\mathcal{G}$  can be arbitrary compounds of other formulas such as  $\mathcal{F}=(A \wedge B)$ and  $\mathcal{F}=\forall x, A(x) \to B(x)$ . Finally, since we can consider

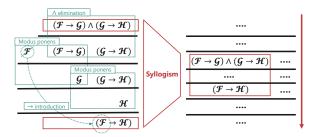


Figure 2: An example of multistep deduction constructed from the axioms. (**Left**) shows the derivation of a syllogism. (**Right**) illustrates that deduction with more steps can express deductions that use a syllogism as a given rule.

infinite patterns of formulas as premises and a conclusion, we have infinite patterns of deduction rules (including both valid and invalid deduction rules).

Next, we consider multistep deductions. Figure 2 shows that syllogism rule can be derived by the multistep deduction constructed from other "atomic" deduction rules. (For other examples, Figure B.5 shows the derivations of the deduction rules used in the previous corpora.) Indeed, in formal logic, there is a set of atomic deduction rules called *the axioms* (listed in Figure B.4a), and the following is known <sup>3</sup>:

**Theorem 2.1** (Completeness of first-order predicate logic (Gödel, 1930)). Any valid <sup>4</sup> deduction rule is derivable by multistep deduction constructed from the axioms. Furthermore, any deduction rule derivable by multistep deduction constructed from the axioms is valid.

Here we have come to the core of formal logic: multistep deduction constructed from the axioms. Thanks to the completeness, all valid deduction rules can be derived in this way. As a consequence, *multistep deduction constructed from the axioms can express multistep deduction constructed from any other deduction rules*, as illustrated in Figure 2 (right).

## 3. Generating Formal Logic Deduction Corpus

The previous deduction corpora (Clark et al., 2021; Betz et al., 2021) used limited or arbitrary sets of deduction rules. However, as we saw in Section 2, the axioms should be the most generalizable to various deduction rules. Thus, we propose a framework named **FLD** (Formal Logic **D**eduction), which generates examples of multistep deduction constructed from the axioms.

Another important feature of **FLD** is that the statements in the examples are constructed randomly except logical structures (i.e., they have no semantics), so that referring to existing knowledge never helps solve the task, but only adhering to deduction rules solves the task. Finally, we designed **FLD** to be highly flexible as in Table 1, so that we

<sup>&</sup>lt;sup>2</sup>A deduction step is invalid when for some truth value assignments, the conclusion is false (=0) even if all the premises are true (=1). See Table B.11b.

<sup>&</sup>lt;sup>3</sup>We limit our focus to first-order predicate logic in this paper.

<sup>&</sup>lt;sup>4</sup>A deduction rule is valid when for all truth value assignments, the conclusion is true (=1) if all the premises are true. See Table B.11a.

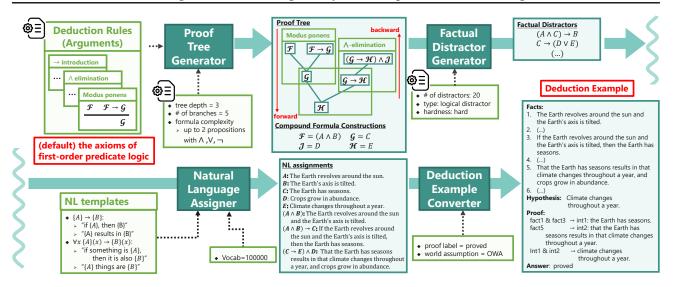


Figure 3: An overview of the proposed framework **FLD**, which generates logical deduction examples constructed from the axioms of first-order predicate logic. **FLD** is flexible to generate various patterns of corpora for analysis. Note that the actual natural language assignments are constructed as random as possible to assess rigid logical reasoning ability isolated from knowledge (see Section 3).

Table 1: A comparison of **FLD** with the previous studies. **FLD** is flexible to generate various patterns of corpora for analysis.  $\checkmark$  means controllable and extensible by an external template file.  $\checkmark$  means controllable by an option.

	Deduction Rules	Proof Tree Depth (upto)	Proof Tree Branches	Formula Complexity	# of Distractors (up to)	Linguistic Diversity	Proof Labels
RuleTaker (Clark et al., 2021)	implication	5	A few	complex	~20	less (RuleTaker) / more (ParaRules)	provable / disprovable / unknown
AACorpus (Betz et al., 2021)	(default = critical thinking)	1	1	√ (simple / complex)	0	(default = less)	provable / disprovable
FLD	(default = the axioms)	(can be any)	(can be any)	√ (simple / complex)	(can be any)	√ (default = more)	(can choose any)

can generate and analyze various patterns of corpora.

We show generated examples in Figures 1 and C.6. Below, we overview each module. For intuitive understanding, refer to the corresponding part of Figure 3. For the detailed implementations, refer to Appendix E.

# 3.1. Proof Tree Generation via Random Forward-/Backward- Deduction

RuleTaker (Clark et al., 2021) generates deductive proof trees by first randomly generating various formulas and second running a logical solver library on them to find occasionally emerged deductive relationships among them. However, since we rely on an external solver, we cannot specify the set of deduction rules used in proof trees (and thus we cannot specify the axioms, especially.). Further, since we rely on the randomness, we cannot control the complexity of a proof tree, i.e., the depth and the number of leaves.

Thus, we decided to take another approach. We invented a

module ("Proof Tree Generator" in Figure 3) that generates a proof tree through a random deduction process by using a set of deduction rules specified by a user. A user can specify the deduction rules in a template rule file, as exemplified in Figure E.7. At each forward- or backward- deduction step, the module randomly chooses one deduction rule and joints it to the current proof tree ("forward" and "backward" in the figure). The numbers of forward- and backward- steps control the tree's depth and number of leaves, respectively.

Once the structure of the proof tree is constructed, we construct the compound formulas at the tree nodes, such as  $\mathcal{F}, \mathcal{G}$ . Since these formulas are arbitrary (Section 2), we randomly combine atomic formulas such as A and B using logical operators  $\land, \lor, \lnot$ . To avoid over complications, we limit the number of atomic formulas in each compound formula up to three. The resulting formulas are like  $\mathcal{F} = (\lnot A \land B)$ .

### 3.2. Factual Distractor Generation

In a realistic scenario of logical reasoning, since the facts are collected by possibly incomplete retrieval systems rather than given, LMs have to correctly choose only the relevant facts under the existence of many irrelevant facts. To imitate this scenario, we add distractor facts to each deduction example ("Factual Distractor Generator" in Figure 3). The distractor facts are formulas that are similar to the gold facts in their logical form. For example, for the gold fact  $(A \wedge B) \to C$ , formulas such as  $(A \wedge C) \to B$  can be distractors. We also implemented several other types of distractors and use the mixture of them.

### 3.3. Natural Language Assignment

We assign one natural language sequence to each formula of tree nodes and of distractors ("Natural Language Assigner" in Figure 3). Inspired by Betz et al. (2021), we take a template based approach. For each formula, we prepare several templates via an external template file (exemplified in Figure E.8) such as follows:

$$A \to B$$
 : "If A, then B.", "A leads to B." 
$$F(a) \to G(b)$$
 : "If a F, then b G.", "When a F, b G."

Then, we randomly choose one from them. Note that since the templates can be nested, the number of resulting patterns are combinatorially diverse.

Next, we assign natural language statements to atomic components such as A, B, F, G, a, b. Here, we come back to the important point in deductive reasoning discussed in Section 2: that the validity of deduction does not depend on contents of formulas, or in other words, the same deduction can be conducted on the same formulas regardless of their contents. To reflect this point, we assign a *random* statement constructed (under a certain grammatical constraint) from a full vocabulary to each atomic component; for example:

```
A: "an Earthquake occurs" B: "the year ends" F: "run" G: "answer" a: "the hamburger" b: "Peter" These randomly constructed statements ensure that referring to existing knowledge never helps solve the task, but only adhering to deduction rules solves the task.
```

## 3.4. Deduction Example Conversion

We finally make a deductive reasoning example from the outputs of the previous modules ("Deduction Example Converter" in Figure 3). A deduction example is composed of a set of facts, a hypothesis, a proof sequence, and an answer ("proved", "disproved", or "unknown"). This module can make an example of any answer label as follows. For answer label "proved", (i) we use the root node as the hypothesis, (ii) we use the leaf nodes of the proof tree and the distractors as the fact set, and (iii) we use the internal nodes of the proof tree as the proof sequence. For answer label

"disproved", we use the negated statement of the root node as the hypothesis so that the hypothesis is disproved by the proof sequence. For answer label "unknown", we randomly drop some of the leaf nodes so that the hypothesis cannot be proved or disproved by the proof sequence.

### 4. Experiments

We conducted experiments to verify the effectiveness of **FLD**, and to identify the aspects of deductive reasoning ability on which deduction corpora can enhance LMs. To this end, we examined various deduction corpora shown in Table 2. We trained LMs on the deduction corpora and measured their performance on relevant benchmarks. For reference, we also measured the performance of a LM (T5) without training on the deduction corpora. We used two types of benchmarks: deduction corpora themselves and human-authored EntailmentBank (Dalvi et al., 2021). We briefly explain the setup. See Appendix F for the details.

#### 4.1. Model

All the experiments involve generating a proof sequence to (dis-)prove a given hypothesis from a given set of facts. To tackle the task of this type, we adopt the stepwise prover model from Yang et al. (2022). This prover is a generative model based on T5 (Raffel et al., 2020), which generates one proof step at a time. A proof step represents the chosen premises and the derived (generated) conclusion, such as "fact1 & fact3 -> The Earth has seasons". The prover continues the generation until the given hypothesis is (dis-)proved. Finally, the prover outputs an answer labeled as "proved," "disproved," or "unknown."

We also evaluated large language models in a few-shot incontext learning setting. Specifically, we evaluated GPT-4, GPT-3.5-Turbo (OpenAI, 2023), and LongAlpaca-13B (Chen et al., 2023). Each in-context example is a pair of a prompt and an output, as illustrated in Figure 1. We also added a chain-of-thought (Kojima et al., 2022) instruction as "Show me a step-by-step thought to the hypothesis based on the given set of facts." We ran experiments for 10 different sets of in-context examples.

#### 4.2. Few-shot Transfer to Synthetic Deduction Corpora

The first benchmark is the deduction corpora, which measures rigid logical reasoning ability. We trained prover LM on a corpus and measured its performance on another corpus. If LMs have acquired robust deductive reasoning ability, they should transfer well with a small number of examples. To see this, we used few-shot setting <sup>5</sup>.

<sup>&</sup>lt;sup>5</sup>Zero-shot is not appropriate for transfer among corpora that differ in the sets of deduction rules used in proofs as follows. Since a proof step is made by a deduction rule, the nature (granularity)

Table 2: The corpora examined in this paper. For RuleTaker ("RT"), we used the OWA version introduced by Tafjord et al. (2021). To align conditions as closely as possible across the corpora being compared, we (i) generated multiple FLD corpora using the options and template files and (ii) added several preprocessings to RuleTaker. See Appendix F.1 for details.

name	deduction rules	distractors (up to)	linguistic diversity	formula complexity	tree depth	tree depth distribution	# train examples
RT ("D0-D3") RT.PR ("ParaRules") RT.BE ("Birds-Electricity")	implication implication implication	$\begin{array}{l} \sim 20 \\ \sim 20 \\ \sim 20 \end{array}$	less more more	complex complex complex	1–3 1–5 1–3	skewed (biased toward	30k 30k - (test only)
sFLD-impl sFLD-crit sFLD-axiom ( <b>sFLD</b> )	implication critical thinking the axioms	$\begin{array}{l} \sim 20 \\ \sim 20 \\ \sim 20 \end{array}$	less less less	complex complex complex	1–3 1–1 1–3	lower depths)	30k 30k 30k
RT.D5 ("D0-D5") FLD.D5	implication the axioms	$\begin{array}{l} \sim 20 \\ \sim 20 \end{array}$	less less	complex complex	1–5 1–5		30k 30k
FLD-impl.0 FLD-impl.1	implication implication	$\begin{array}{l} \sim 20 \\ \sim 20 \end{array}$	less less	complex complex	1–3 1–8	uniform	30k 30k
FLD.0 FLD.1 FLD.2 FLD.3 (FLD) FLD.4 (FLD*)	the axioms the axioms the axioms the axioms the axioms	$0 \sim 20 \sim 20 \sim 20 \sim 20 \sim 20 \sim 20$	less less less more more	complex simple complex complex complex	1-3 1-3 1-3 1-3 1-8		30k 30k 30k 30k 30k 30k

We measure the performance of the prover on the test split of the target corpus using two types of metrics, answer accuracy and proof accuracy (Saha et al., 2020). Answer accuracy measures whether predicted answers, each of which is chosen from "proved", "disproved" and "unknown", are correct or not. Proof accuracy measures whether *both* the predicted answers and the generated proofs are correct or not, offering a stricter and more faithful assessment of reasoning ability. Our proof accuracy metric is stricter than the original by Saha et al. (2020) (details in our repository<sup>1</sup>). We focus our discussion on proof accuracy results, with answer accuracy results presented in Appendix G.1.

We trained prover LM (T5-base) on the training split of each source corpus for 20k steps with a batch size of 64 and learning rate of 1e-4. Then we fine-tuned the prover LM on 1% subset (300 examples) of the training split of the target corpus.

## 4.3. Transfer to EntailmentBank

EntailmentBank (EB) (Dalvi et al., 2021) is a recently proposed challenging benchmark. The proof trees in the EB dataset are human-authored rather than synthetically generated. Further, each proof step can be rough entailment instead of a rigid logical step. Thus, EB measures logical reasoning ability in a more real-world scenario.

We used all the three tasks of EB, which differ in the property of a given fact set: Task1 does not include distractors, Task2 includes distractors, and Task3 includes sentences retrieved from worldTree V2 (Xie et al., 2020).

of proof steps in one corpus differs much from those in another corpus. To adjust this artificial difference, LMs need examples of the target corpus.

As stated above, the nature of proof steps in EB differs much from the nature of those in deduction corpora. Thus, it is difficult for prover LMs trained on deduction corpora to transfer to EB with a small number of examples. Thus, we fine-tuned the provers using all the EB examples.

We trained a prover LM (T5-large) on a source deduction corpus for 10k steps and fine-tuned it on each EB corpus for 10k steps. For all the training, the batch size was 64 and the learning rate was 5e-5, except EB-task2 where the learning rate of 2.5e-5 was used. For EB-task3, we used the prover trained on task2, following Dalvi et al. (2021). Given the challengingness of EB, we used the additional RoBERTa (Liu et al., 2019) based proof step verifier proposed in Yang et al. (2022). We measured the performance of the provers on the test split of EB by the official metric of "AllCorrect" proof accuracy (Dalvi et al., 2021).

## 5. How Well do LMs Solve Logic?

### 5.1. Performance of LMs in Fine-tuned Setting

Table 3: Proof accuracy of a prover fully fine-tuned using all the dataset examples on each corpus. Note that we released<sup>1</sup> the updated, version 2 of **FLD** corpora, on which the provers perform slightly better (See Appendix H for details).

Rule	Taker	FLD		
RT	RT RT.PR		FLD∗	
92.4	93.9	66.4	37.7	

First, we show how well LMs solve logic of each deduction corpus (Table 3). As shown, while the fully fine-tuned provers performed well on RuleTaker, they performed poorer on FLD. One possible reason is as follows. First, since a proof tree is constructed from the combination of a

deduction rule chosen at each level of the tree, the number of possible proof tree patterns can be estimated (very roughly) as  $\mathcal{O}(\mathcal{A}^d)$ , where  $\mathcal{A}$  is the number of deduction rule choices and d is the proof tree depth. Next, while RuleTaker uses only a few deduction rules ( $\mathcal{A}=2$ ) of implication type shown in Figure B.4b, FLD uses various deduction rules ( $\mathcal{A}\sim10$ ) of the axioms shown in Figure B.4a. Thus, FLD includes exponentially more diverse patterns of proof trees, which makes FLD more challenging. Indeed, when we enlarge the maximum tree depth from d=3 to d=8 (FLD to FLD\*), the corpus became extremely more challenging due to the exponentially more diverse proof tree patterns. See Appendix G for further detailed analysis.

## 5.2. Performance of LLMs in Few-shot Setting

Table 4: Performances of LLMs in a 10-shot in-context learning setting. Chain-of-thought-like instruction was also used. For reference, the performances of random guesses and T5 fine-tuned on all the 30,000 examples are also shown. We used the version 2 corpora (See Appendix H).

	proof a	accuracy	answer accuracy		
	FLD	FLD⋆	FLD	FLD*	
random guess	0.0	0.0	33.3	33.3	
T5(fine-tuned)	75.8	44.4	91.6	72.2	
LongAlpaca-13B	0.0	0.0	21.2	19.6	
GPT-3.5-Turbo	0.0	2.0	35.8	37.6	
GPT-4	12.8	3.2	52.4	49.4	

Table 4 show the performance of LLMs measured under 10-shot settings. As seen from the proof accuracy results, even the most powerful LLM, GPT-4, performed very poorly. Further, even when measured by answer accuracy, which is more lenient than proof accuracy, GPT-4 solved only half of the problems.

The challengingness of **FLD** could be attributed to its counterfactual nature. The facts in the deduction examples are randomly constructed except for logical structure (i.e., they have no semantics). Due to this nature, an LLM can *never* rely on its pre-acquired knowledge to solve the task, but it has to adhere to the deduction rules. Such a kind of task should not be covered by pre-training on human-written texts. The poor performance of LLMs under the counterfactual setting is consistent with the recent observations by (Razeghi et al., 2022; Wu et al., 2023; Hodel & West, 2023; Dasgupta et al., 2023).

As can be seen, the answer accuracy performance deviated much from the proof accuracy performance. This gap can be partly explained by the "random guess factor", but not entirely. We analyzed the proofs generated by the LLMs and found that *they sometimes generated a correct answer* 

with an incorrect proof. This suggests that the LLMs do not always faithfully follow the "logical steps" they themselves generate. These results align with recent findings by Turpin et al. (2023); Lanham et al. (2023).

Finally, we manually analyzed and categorized the common errors of GPT-4, as follows. The first is *logical hallucination*, where the generated conclusion is not logically deducible from the chosen facts. Second, GPT-4 sometimes chooses facts from distractors that are irrelevant to the proof. Taking a closer look, GPT-4 occasionally misinterprets logical operators, especially double-negation ¬¬. Interestingly, GPT-4 seldom answered "unknown", i.e., it answered "proved" or "disproved" even if the given facts are not sufficient to either prove or disprove the hypothesis. Overall, these errors suggest that LLMs still miss the fundamentals of logical reasoning.

## 6. How Effective is Formal Logic Deduction?

### 6.1. Benchmarking by Deduction Corpora

Table 5: Few-shot proof accuracies of provers transferred among **sFLD** and baseline corpora. For fair comparison, all the corpora have the same depth distribution (except sFLD-crit that cannot form multistep easily, see Appendix F.1)

			Source corpus					
		T5	RT	RT.PR	sFLD-impl	sFLD-crit	sFLD	
	RT	70.1	92.4	91.3	76.2	74.4	76.7	
	RT.PR	64.3	91.3	93.9	73.4	67.5	72.9	
Target	RT.BE	56.1	88.3	88.2	75.2	79.4	85.0	
corpus	sFLD-impl	58.4	66.7	65.9	82.2	67.3	80.7	
	sFLD-crit	71.9	77.7	77.2	87.8	94.0	93.6	
	sFLD	54.7	54.5	54.5	67.9	63.7	79.1	
	avg.	62.6	78.5	78.5	77.1	74.4	81.3	

We trained a prover on a deduction corpus ("source corpus") and measured its performance on other corpora ("target corpus") (Table 5). The prover trained on **SFLD** performed the best on average, and as seen from the corpus-wise results, the prover transferred the most robustly to the other corpora while the provers trained on the other corpora did not exhibit this level of robustness. Since the corpora used in Table 5 differ in the set of deduction rules used in proofs, this result suggests that the prover trained **SFLD** generalized the most to other deduction rules.

The reason for this strongest generalizability should be the following. (s)FLD corpora teach LMs how to construct multistep deductions using the axioms. Thanks to the completeness, the axioms can express multistep deductions constructed from any other deduction rules (including the ones used in the other corpora, as exemplified in Figure B.5). Thus, mastering the axioms leads to mastering various other deduction rules. On the other hand, the sets of deduction rules used in the other corpora do not have such a property and thus cannot be generalized to other deduction rules.

Since mastering various deduction rules is the most important in deductive reasoning, this generalizability to deduction rules obtained from **FLD** corpora is vital.

### 6.2. Benchmarking by EntailmentBank

Table 6: The proof accuracy of provers on EntailmentBank. See Appendix G.2 for the results of other metrics.

		EntailmentBank			
		Task1	Task2	Task3	
Source	T5 RT.D5	$36.8_{\pm 0.9}$ $39.4_{\pm 0.9}$	$31.2_{\pm 0.7}$ $32.0_{\pm 0.8}$	$6.2_{\pm 0.9}$ $8.2_{\pm 0.8}$	
corpus	FLD.D5	$39.2_{\pm 1.2}$	$32.0_{\pm 0.8}$ $32.6_{\pm 1.0}$	$8.2_{\pm 0.8}$ $8.3_{\pm 0.7}$	

Table 6 shows the results on EntailmentBank (EB). Since EB trees have high-depth (majority up to five), we used the high-depth versions of deduction corpora as source corpus.

First, as seen, the provers trained on both deduction corpora (RT.D5, **FLD.D5**) performed better than the baseline prover without such training (T5). This suggests that the deductive reasoning ability acquired by synthetic deduction corpora generalizes to more complex real-world deductive reasoning. We showcase some examples in Appendix G.3, where the error of the baseline prover is fixed by training on a deduction corpus (**FLD.D5**). As seen, the prover captured the fundamentals of deduction rules better than the baseline as follows: (i) it chose the correct premises necessary and sufficient to derive the next conclusion, (ii) it included only such information that logically follows from the chosen premises into a conclusion, and (iii) and it correctly used the rules of logical operators.

Looking at the results of deduction corpora closely, the prover trained on **FLD.D5** performed on par with the prover trained on RT.D5, even though it had mastered various deduction rules better, as shown in Section 6.1. We consider a possible reason as follows. Firstly, real-world reasoning can require more coarse-grained deduction rules than those required by deduction corpora. For expressing such coarse-grained deduction rules by the most fine-grained axioms, many steps are required, as in Figure 2. However, the prover trained on **FLD** still struggles with constructing many-step proofs using the axioms (detailed in Section 7.1). In this sense, the prover could have failed to exploit the axioms' potential fully. We will discuss future directions to tackle this challenge in Section 8.

# 7. On What Aspects are Synthetic Deduction Corpora Beneficial?

A deduction corpus in Table 2 emphasizes a specific aspect different from those emphasized by the other corpora. For each corpus (each aspect), we investigate whether the LM trained on that corpus outperforms the LM trained on the other corpus that does not emphasize the aspect. If it does, we interpret it as meaning that the supervision from deduction corpus on that aspect is beneficial for LMs.

### 7.1. Ability to Solve Complex Proof Trees

Table 7: The depth-wise proof accuracies of the provers.

(a) Target corpus is FLD-impl.1 (b) Target corpus is FLD.4

		Source corpus					
	T5	FLD-impl.0	FLD-impl.1				
0	41.7	83.3	70.8				
1	77.4	88.7	93.5				
2	38.0	56.0	53.0				
3	33.8	50.0	47.5				
4	36.7	51.1	40.0				
5	21.4	42.9	42.9				
6	22.7	38.7	41.3				
7	25.5	38.7	44.3				
8	23.0	36.1	37.7				
avg.	35.6	53.9	52.3				

	Source	Source corpus					
T5	FLD.3	FLD.4					
75.0	100.0	100.0					
64.9	98.6	95.9					
2.3	62.5	59.1					
1.1	18.9	23.7					
0.0	1.4	7.5					
0.0	0.9	2.3					
0.0	0.0	0.0					
0.0	0.0	0.0					
0.0	0.0	1.2					
15.9	31.4	32.2					

Table 7 shows the depth-wise performances of provers. The corpora in Table 7a use the implication deduction rules. The prover trained on the corpus of shallower ( $\sim$  3) trees (FLD-impl.0) generalizes to deeper ( $4\sim$ 8) trees to some extent, and performs similarly to the prover trained on the corpus of deeper trees (FLD-impl.1). This generalization to deeper trees coincides with previous findings (Tafjord et al., 2021; Sanyal et al., 2022). However, as Table 7b shows, when the corpora use the axioms, neither the provers trained on the shallower tree corpus (FLD.3) nor deeper tree corpus (FLD.4) failed in solving deeper trees.

We can interpret this seemingly contradictory result as follows. As discussed in Section 5, the number of possible proof tree patterns can be estimated (very roughly) as  $\mathcal{O}(\mathcal{A}^d)$ . When a prover tries to solve a deduction example, it has to choose and generate exactly the one gold proof tree from these possible negative proof trees. This should be very difficult for large d with large  $\mathcal{A}$ . Now, while the corpora in Table 7a use a few deduction rules ( $\mathcal{A}=2$ ) of implication type, corpora in Table 7b use various deduction rules ( $\mathcal{A}\sim 10$ ) of the axioms. This made it very difficult to solve large-depth deduction examples of these corpora, which lead the provers to fail in solving large-depth proof trees in Table 7b.

Overall, for solving complex trees, the supervision from deduction corpora can be necessary but not sufficient alone.

### 7.2. Understanding of Diverse Linguistic Expressions

Table 8: Few-shot proof accuracies of provers transferred among corpora that differ in the diversity of linguistic expressions.

			Source corpus			
			Rule	RuleTaker		_D
		T5	RT	RT.PR	FLD.2	FLD.3
	RT	70.1	92.4	91.3	78.3	76.6
Target	RT.BE	64.3	91.3	93.9	71.3	73.4
corpus	FLD.2	31.0	34.2	34.7	66.8	66.2
	FLD.3	24.8	28.7	27.5	65.3	66.4
	avg.	47.6	61.6	61.8	70.4	70.7

Table 8 shows that a prover trained on a corpus with less linguistic diversity (i.e., RT and FLD.2) performed as well as the prover trained on the linguistically diverse counterpart of that corpus (i.e., RT.PR and FLD.3, respectively). This suggests that LMs are self-sufficient on the linguistic aspect, and thus additional supervision from deduction corpora is not that important.

Indeed, this result coincides with the previous findings (Clark et al., 2021; Tafjord et al., 2021) and can be intuitively understood: since the pre-training corpora of LMs are huge and linguistically diverse, they should have given LMs many chances to learn linguistic of logical statements such as that "If A, then B" paraphrases to "A leads to B".

### 7.3. Understanding of Complex Formulas

Table 9: Few-shot proof accuracies of provers transferred among corpora that differ in the complexity of formulas.

			Source corpus		
		T5	FLD.1	FLD.2	
Target	FLD.1	43.1	77.9	71.6	
corpus	FLD.2	31.0	46.0	66.8	

Table 9 shows that while the prover trained on the corpus with simple formulas (FLD.1) performed poorly on the corpus with complex formulas (FLD.2), the prover trained FLD.2 performed well on both corpora. Thus, deduction corpora are beneficial for mastering complex formulas.

We can interpret this result as follows. The complex formulas included in FLD.2 are formed by modifying atomic formulas with logical operators  $\neg$ ,  $\land$ ,  $\lor$ . The semantics of these logical operators, such that "a sentence with negation  $\neg$  have the opposite meaning of that sentence without negation", and that " $A \lor B$  does not necessarily imply A", are seldom written explicitly by humans. Thus, the pre-training corpora gave LMs too few chances for learning these semantics. This result is enhanced by the previous findings that LMs fail to understand the semantics of negation (Naik et al., 2018; Hossain et al., 2020; Kassner & Schütze, 2020).

### 7.4. Robustness to Distractive Facts

Table 10: Few-shot proof accuracies of provers transferred among corpora that differ in the number of distractors.

			Source corpus		
		T5	FLD.0	FLD.2	
Target	FLD.0	38.9	76.4	75.2	
corpus	FLD.2	31.0	56.7	66.8	

Table 10 shows that, while the prover trained on the corpus without distractors (FLD.0) performed poorly on the corpus with distractors (FLD.2), the prover trained on FLD.2 performed well on both corpora. Thus, synthetic distractors are beneficial for acquiring the robustness to distractive facts. This result is intuitive: since the human-written text

should not include the facts irrelevant to the content, the pre-training corpora should not have given LMs a chance to acquire robustness to irrelevant facts.

### 8. Discussions and Future Directions

So far, we have investigated each aspect of deductive reasoning. We summarize the results and discuss future directions.

Mastery on Various Deduction Rules: Mastering various deduction rules is the most important in deductive reasoning. We showed that **FLD** corpora teach LMs various deduction rules the most effectively (Section 6.1). This should be because that **FLD** adopts the axioms of first-order predicate logic system, which can derive any valid deduction rules in this system. The next step will be to examine the axioms of other logic systems, such as linear and modal logic systems, which are also important in real-world reasoning.

Ability to Solve Complex Proof Trees: We have shown that solving a many-step proof tree is still challenging for LMs even after training on deduction corpora (Section 7.1). The possible reason is that they have to choose and generate a gold proof from a large number of possible trees. To solve this problem, inventing smarter and strategic search methods on possible generation space, such as Li et al. (2016); Negrinho et al. (2018); Picco et al. (2021); Welleck et al. (2022), could be a promising direction.

Understanding of Complex Formulas: We have shown that deduction corpora are effective for LMs to understand the semantics of logical operators such as  $\neg$ ,  $\wedge$ ,  $\vee$  (Sections 6.2 and 7.3). It could be even more effective to incorporate the recent learning methodological approaches for making LMs understand negation (Pröllochs et al., 2019; Hosseini et al., 2021) into the learning on deduction corpora.

Robustness to Distractive Facts: We have shown that the synthetic distractors can make LMs robust to distractive facts (Section 7.4). In a real scenario of logical reasoning, the facts have to be collected by possibly incomplete retrieval systems. The distractors that imitate ones appearing in such a scenario could be more effective. We can generate such distractors as follows: (i) We build a database of synthetic facts. (ii) For a given deduction example, we collect facts from the database by actual retrieval systems.

Generalization to Real-World Reasoning Tasks: We have shown that the training on deduction corpora is even useful for deductive reasoning in a more real-world setting (Section 6.2). However, the LMs trained on **FLD** could not fully utilize the potential of the axioms, as they failed in constructing many-step proofs to express coarse-grained deduction rules, which could be required in real-world reasoning (Sections 6.2 and 7.1). We discussed future directions to solve such many-step proofs above.

Further, LMs may need additional training to utilize deduction rules well in a realistic context. For example, the LMs

could have to combine deduction rules with common sense knowledge, use multiple deduction rules at once to jump to the next conclusion, and judge the validity of a proof step considering the overall context. Recently, Wei et al. (2022); Kojima et al. (2022) showed that large LMs can utilize deduction rules in a realistic context, given appropriate prompts. It could be promising to integrate this approach and deduction corpora training.

Pursuing further real-world scenarios, we have to tackle tasks of other settings. One is deductive reasoning that requires us to collect relevant facts by ourselves. For this, we could exploit factual knowledge implicitly embedded in LMs (Petroni et al., 2019; Davison et al., 2019; Talmor et al., 2020), or use retrieval systems. For the latter, we could train LM-based retrievers (Karpukhin et al., 2020; Guu et al., 2020) using synthetic deduction examples and fact database. Abductive reasoning (Bhagavatula et al., 2019) is another kind of real-world logical reasoning with which we derive hidden premises from a conclusion and other visible premises. Synthetic corpora for abduction based on formal logic can be generated similarly to as done in this study.

#### 9. Conclusion

To teach language models deductive reasoning, we proposed a synthetic corpus based on formal logic theory and verified its effectiveness empirically. Further, we analyzed each aspect of deductive reasoning and provided future directions on each. We will advance on the basis of these directions.

## Acknowledgement

We thank the three anonymous reviewers and the metareviewer, who gave us insightful comments and suggestions. Computational resources of AI Bridging Cloud Infrastructure (ABCI) provided by the National Institute of Advanced Industrial Science and Technology (AIST) were used. We thank Dr. Masaaki Shimizu at Hitachi for the convenience of additional computational resources. We thank Dr. Naoaki Okazaki, professor at Tokyo Institute of Technology, for the keen comments.

### References

- Askell, A. Gpt-3: Towards renaissance models. *Daily Nous Blog: Philosophers On GPT-3*, 2020.
- Betz, G., Voigt, C., and Richardson, K. Critical thinking for language models. In *Proceedings of the 14th International Conference on Computational Semantics (IWCS)*, pp. 63–75, Groningen, The Netherlands (online), June 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.iwcs-1.7.
- Bhagavatula, C., Bras, R. L., Malaviya, C., Sakaguchi, K.,

- Holtzman, A., Rashkin, H., Downey, D., Yih, S. W.-t., and Choi, Y. Abductive commonsense reasoning. *arXiv* preprint arXiv:1908.05739, 2019.
- Bostrom, K., Zhao, X., Chaudhuri, S., and Durrett, G. Flexible generation of natural language deductions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6266–6278, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.506. URL https://aclanthology.org/2021.emnlp-main.506.
- Bostrom, K., Sprague, Z., Chaudhuri, S., and Durrett, G. Natural language deduction through search over statement compositions. *CoRR*, abs/2201.06028, 2022. URL https://arxiv.org/abs/2201.06028.
- Chen, Y., Qian, S., Tang, H., Lai, X., Liu, Z., Han, S., and Jia, J. Longlora: Efficient fine-tuning of long-context large language models. *arXiv preprint arXiv:2309.12307*, 2023.
- Clark, P., Tafjord, O., and Richardson, K. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 3882–3890, 2021.
- Dalvi, B., Jansen, P., Tafjord, O., Xie, Z., Smith, H., Pipatanangkura, L., and Clark, P. Explaining answers with entailment trees. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 7358–7370, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main. 585. URL https://aclanthology.org/2021.emnlp-main.585.
- Dasgupta, I., Lampinen, A. K., Chan, S. C. Y., Sheahan, H. R., Creswell, A., Kumaran, D., McClelland, J. L., and Hill, F. Language models show human-like content effects on reasoning tasks, 2023.
- Davison, J., Feldman, J., and Rush, A. Commonsense knowledge mining from pretrained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 1173–1178, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1109. URL https://aclanthology.org/D19-1109.
- Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *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), pp. 4171–4186, 2019.
- Dziri, N., Lu, X., Sclar, M., Li, X. L., Jiang, L., Lin, B. Y., West, P., Bhagavatula, C., Bras, R. L., Hwang, J. D., Sanyal, S., Welleck, S., Ren, X., Ettinger, A., Harchaoui, Z., and Choi, Y. Faith and fate: Limits of transformers on compositionality, 2023.
- Gödel, K. *Uber die vollständigkeit des logikkalküls*. PhD thesis, Ph. D. dissertation, University of Vienna, 1930.
- Gontier, N., Sinha, K., Reddy, S., and Pal, C. Measuring systematic generalization in neural proof generation with transformers. Advances in Neural Information Processing Systems, 33:22231–22242, 2020.
- Guu, K., Lee, K., Tung, Z., Pasupat, P., and Chang, M. Retrieval augmented language model pre-training. In *International Conference on Machine Learning*, pp. 3929–3938. PMLR, 2020.
- Habernal, I., Wachsmuth, H., Gurevych, I., and Stein, B. The argument reasoning comprehension task: Identification and reconstruction of implicit warrants. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1930–1940, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1175. URL https://aclanthology.org/N18-1175.
- Han, S., Schoelkopf, H., Zhao, Y., Qi, Z., Riddell, M., Benson, L., Sun, L., Zubova, E., Qiao, Y., Burtell, M., Peng, D., Fan, J., Liu, Y., Wong, B., Sailor, M., Ni, A., Nan, L., Kasai, J., Yu, T., Zhang, R., Joty, S., Fabbri, A. R., Kryscinski, W., Lin, X. V., Xiong, C., and Radev, D. Folio: Natural language reasoning with first-order logic, 2022. URL https://arxiv.org/abs/2209.00840.
- Hodel, D. and West, J. Response: Emergent analogical reasoning in large language models, 2023.
- Hossain, M. M., Kovatchev, V., Dutta, P., Kao, T., Wei, E., and Blanco, E. An analysis of natural language inference benchmarks through the lens of negation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 9106–9118, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main. 732. URL https://aclanthology.org/2020.emnlp-main.732.

- Hosseini, A., Reddy, S., Bahdanau, D., Hjelm, R. D., Sordoni, A., and Courville, A. Understanding by understanding not: Modeling negation in language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1301–1312, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main. 102. URL https://aclanthology.org/2021.naacl-main.102.
- Kahneman, D. *Thinking, fast and slow*. Macmillan, 2011.
- Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., and Yih, W.-t. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, 2020.
- Kassner, N. and Schütze, H. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7811–7818, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main. 698. URL https://aclanthology.org/2020.acl-main.698.
- Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners, 2022. URL https://arxiv.org/abs/2205.11916.
- Lanham, T., Chen, A., Radhakrishnan, A., Steiner, B., Denison, C., Hernandez, D., Li, D., Durmus, E., Hubinger, E., Kernion, J., Lukošiūtė, K., Nguyen, K., Cheng, N., Joseph, N., Schiefer, N., Rausch, O., Larson, R., McCandlish, S., Kundu, S., Kadavath, S., Yang, S., Henighan, T., Maxwell, T., Telleen-Lawton, T., Hume, T., Hatfield-Dodds, Z., Kaplan, J., Brauner, J., Bowman, S. R., and Perez, E. Measuring faithfulness in chain-of-thought reasoning, 2023.
- Li, J., Monroe, W., Ritter, A., Jurafsky, D., Galley, M., and Gao, J. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1192–1202, 2016.
- Lin, K., Tafjord, O., Clark, P., and Gardner, M. Reasoning over paragraph effects in situations. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pp. 58–62, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-5808. URL https://aclanthology.org/D19-5808.

- Liu, H., Ning, R., Teng, Z., Liu, J., Zhou, Q., and Zhang, Y. Evaluating the logical reasoning ability of chatgpt and gpt-4, 2023.
- Liu, J., Cui, L., Liu, H., Huang, D., Wang, Y., and Zhang, Y. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. In Bessiere, C. (ed.), Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20, pp. 3622–3628. International Joint Conferences on Artificial Intelligence Organization, 7 2020. doi: 10.24963/ijcai.2020/501. URL https://doi.org/10.24963/ijcai.2020/501. Main track.
- Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. RoBERTa: A robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692, 2019.
- McCarthy, J. W. Programs with common sense. In *Proc. Tedding Conf. on the Mechanization of Thought Processes*, pp. 75–91, 1959.
- Mishra, B. D., Tafjord, O., and Clark, P. Towards teachable reasoning systems: Using a dynamic memory of user feedback for continual system improvement, 2022. URL https://arxiv.org/abs/2204.13074.
- Naik, A., Ravichander, A., Sadeh, N., Rose, C., and Neubig, G. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 2340–2353, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://aclanthology.org/C18-1198.
- Negrinho, R., Gormley, M., and Gordon, G. J. Learning beam search policies via imitation learning. *Advances in Neural Information Processing Systems*, 31, 2018.
- OpenAI. Gpt-4 technical report. *ArXiv*, abs/2303.08774, 2023. URL https://api.semanticscholar.org/CorpusID:257532815.
- Petroni, F., Rocktäschel, T., Riedel, S., Lewis, P., Bakhtin, A., Wu, Y., and Miller, A. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2463–2473, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1250. URL https://aclanthology.org/D19-1250.
- Picco, G., Hoang, T. L., Sbodio, M. L., and Lopez, V. Neural unification for logic reasoning over natural language. In Findings of the Association for Computational Linguistics: EMNLP 2021,

- pp. 3939–3950, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp. 331. URL https://aclanthology.org/2021.findings-emnlp.331.
- Pröllochs, N., Feuerriegel, S., and Neumann, D. Learning interpretable negation rules via weak supervision at document level: A reinforcement learning approach. In *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)*, pp. 407–413, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1038. URL https://aclanthology.org/N19-1038.
- Rae, J. W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, F., Aslanides, J., Henderson, S., Ring, R., Young, S., et al. Scaling language models: Methods, analysis & insights from training gopher. arXiv preprint arXiv:2112.11446, 2021.
- Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P. J., et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.
- Razeghi, Y., Logan IV, R. L., Gardner, M., and Singh, S. Impact of pretraining term frequencies on few-shot numerical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 840–854, 2022.
- Richardson, K., Hu, H., Moss, L., and Sabharwal, A. Probing natural language inference models through semantic fragments. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8713–8721, 2020.
- Saha, S., Ghosh, S., Srivastava, S., and Bansal, M. PRover: Proof generation for interpretable reasoning over rules. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 122–136, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main. 9. URL https://aclanthology.org/2020.emnlp-main.9.
- Sanyal, S., Singh, H., and Ren, X. Fairr: Faithful and robust deductive reasoning over natural language. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1075–1093, 2022.
- Sun, C., Zhang, X., Chen, J., Gan, C., Wu, Y., Chen, J., Zhou, H., and Li, L. Probabilistic graph reasoning for natural proof generation. In *Findings of the Association for*

- Computational Linguistics: ACL-IJCNLP 2021, pp. 3140–3151, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl. 277. URL https://aclanthology.org/2021.findings-acl.277.
- Tafjord, O., Gardner, M., Lin, K., and Clark, P. Quartz: An open-domain dataset of qualitative relationship questions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 5941–5946, 2019.
- Tafjord, O., Dalvi, B., and Clark, P. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3621–3634, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl. 317. URL https://aclanthology.org/2021.findings-acl.317.
- Talmor, A., Tafjord, O., Clark, P., Goldberg, Y., and Berant, J. Leap-of-thought: Teaching pre-trained models to systematically reason over implicit knowledge. *Advances in Neural Information Processing Systems*, 33: 20227–20237, 2020.
- Tian, J., Li, Y., Chen, W., Xiao, L., He, H., and Jin, Y. Diagnosing the first-order logical reasoning ability through logicnli. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3738–3747, 2021.
- Turpin, M., Michael, J., Perez, E., and Bowman, S. R. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting, 2023.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.
- Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Black-boxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355, 2018.
- Wei, J., Wang, X., Schuurmans, D., Bosma, M., brian ichter, Xia, F., Chi, E. H., Le, Q. V., and Zhou, D. Chain of thought prompting elicits reasoning in large language models. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), Advances in Neural Information Processing Systems, 2022. URL https://openreview.net/forum?id=\_VjQlMeSB\_J.

- Welleck, S., Liu, J., Lu, X., Hajishirzi, H., and Choi, Y. Naturalprover: Grounded mathematical proof generation with language models. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=rhdfTOiXBng.
- Weston, J., Bordes, A., Chopra, S., Rush, A. M., Van Merriënboer, B., Joulin, A., and Mikolov, T. Towards aicomplete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*, 2015.
- Wu, Z., Qiu, L., Ross, A., Akyürek, E., Chen, B., Wang, B., Kim, N., Andreas, J., and Kim, Y. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks, 2023.
- Xie, Z., Thiem, S., Martin, J., Wainwright, E., Marmorstein, S., and Jansen, P. WorldTree v2: A corpus of science-domain structured explanations and inference patterns supporting multi-hop inference. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pp. 5456–5473, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.671.
- Yang, K., Deng, J., and Chen, D. Generating natural language proofs with verifier-guided search. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022.

### A. Related Work

# A.1. Synthetic Corpus for Acquiring Deductive Reasoning Ability

A synthetic deduction corpus can be one promising approach for language models (LMs) to acquire logical deductive reasoning ability. The automatic (programmatic) generation ensures the validity of the resulting deductive proof examples. Further, since we can bypass high-cost human annotations we can generate many examples, which should be required by LMs to learn deductive reasoning inductively.

RuleTaker (Clark et al., 2021) proposed a deduction corpus composed of synthetically generated multistep deductive proofs written in natural languages. Each deductive proof (dis-)proves a hypothesis by applying deduction rules multiple times to a given set of facts. As the deduction rules, rules of the implication kind were used. They showed that Transformer (Vaswani et al., 2017) LMs can solve these problems in the sense that they can predict the final answer (i.e., "proved", "disproved", or "unknown") of each deductive proof given the fact set. Later studies (Saha et al., 2020; Dalvi et al., 2021; Tafjord et al., 2021; Sanyal et al., 2022) showed that generative LMs can generate even the intermediate proofs as well as the final answer.

Artificial Argument Corpus (Betz et al., 2021) is another corpus composed of synthetically generated single-step deductive proofs. As the deduction rules, hand-selected rules useful for critical thinking were used. They showed that the LMs trained on this corpus not only solve the task of this corpus itself but generalize to other NLI tasks from GLUE benchmark (Wang et al., 2018). However, at the same time, they showed that such LMs do not generalize well to more challenging logical reasoning tasks such as ARC (Habernal et al., 2018) and LogiQA (Liu et al., 2020).

Gontier et al. (2020) investigated the deductive reasoning ability of LMs on a corpus, which is composed of a specific type of multistep inference: kinship relationship on synthetic kinship graphs. They found that LMs can solve this task when the number of proof steps is not large, but it is difficult for them to generalize to longer-than-training proofs. Bostrom et al. (2021) studied how we can create realistic natural language expressions that represent deduction rules. To this end, they scraped sentences from Wikipedia by a template-based method and paraphrased them. They showed that training on this corpus is helpful for solving real-world deductive reasoning such as EntailmentBank (Dalvi et al., 2021).

While all these corpora focused on specific sets of deduction rules, we focus on the theoretically well-grounded set of deduction rules that can derive any other deduction rules. Further, we analyze each aspect of deductive reasoning using corpora of various patterns to advance the research direction of deductive reasoning.

## A.2. Benchmarks for Deductive Reasoning

Many benchmarks of single-step logical reasoning using specific reasoning rules have been proposed: bAbI (Weston et al., 2015), QuaRTz (Tafjord et al., 2019), ROPES (Lin et al., 2019) and Richardson et al. (2020). For multistep deductive reasoning, FOLIO (Han et al., 2022) is a human-authored benchmark of the SAT (i.e., satisfiability) problem. Given a set of facts and hypotheses, which are created by a human referencing a specific page of Wikipedia, we are required to assign a truth value to each hypothesis. This requires (implicitly) conducting multistep deductive reasoning using high-granularity deduction rules.

RuleTaker (Clark et al., 2021; Tafjord et al., 2021) can work as a benchmark as well as a synthetic training corpus. Rule-Taker focuses on multistep deductive reasoning constructed from implication rules. Since RuleTaker requires generating all the intermediate steps as well as the final prediction on the hypothesis, it is suitable for measuring deductive reasoning ability explicitly and transparently. Further, it includes many irrelevant facts so that the model has to choose only relevant facts under these noises. This makes the task challenging. LogicNLI (Tian et al., 2021) can be considered as an extension of RuleTaker, where additional logical operators such as "\(\equiv\)" are used \(^6\). Additionally, the examples of LogicNLI are checked manually by humans to ensure their quality.

EntailmentBank (EB) (Dalvi et al., 2021) is the same type of task as RuleTaker, but is even more challenging. The proof trees in EB dataset are human-authored rather than synthetically generated. Further, each proof step can be a rough entailment instead of a rigid logical step. Thus, EB measures logical reasoning ability in a more real-world setting.

### A.3. Proof Generation Models

Earlier work (Saha et al., 2020; Gontier et al., 2020; Dalvi et al., 2021; Sun et al., 2021) generated proof sequences at once by LMs. Later work (Tafjord et al., 2021; Bostrom et al., 2022; Sanyal et al., 2022; Yang et al., 2022) generated proofs step-wisely one by one. The stepwise methods are more faithful and robustly generalize to longer proofs. Recently, Wei et al. (2022); Kojima et al. (2022) showed that large language models (LLMs) perform well on multi-hop inference tasks provided appropriate prompts. However, Yang et al. (2022) showed that LLMs given few-shot examples perform poorer than fine-tuned smaller LMs.

<sup>&</sup>lt;sup>6</sup>We do not compare our method with this study directly because the code and corpora are not publicly available and the paper does not clarify the exact rules used.

Table B.11: Truth values of the premises ( $\mathcal{P}_i$ ) and the conclusion ( $\mathcal{C}$ ) of the two deduction rules. Blue shows truth value assignments where both the premises and the conclusion are true (=1). Red shows truth value assignments where the conclusion is false even if all the premises are true.

(a) The valid deduction rule (4) (syllo- (b) The invalid deduction gism).

$$\mathcal{P}_1 = (\mathcal{F} \to \mathcal{G}) \land (\mathcal{G} \to \mathcal{H}),$$
  
 $\mathcal{C} = \mathcal{F} \to \mathcal{H}.$ 

$\overline{\mathcal{F}}$	$\mathcal{G}$	$\mathcal{H}$	$ \mathcal{P}_1 $	С
1	1	1	1	1
1	1	0	0	0
1	0	1	0	1
1	0	0	0	0
0	1	1	1	1
0	1	0	0	1
0	0	1	1	1
0	0	0	1	1

rule (6). 
$$\mathcal{P}_1 = \mathcal{F}, \mathcal{P}_2 = \mathcal{F} \vee \mathcal{G}, \ \mathcal{C} = \mathcal{G}.$$

$\overline{\mathcal{F}}$	$\mathcal{G}$	$ \mathcal{P}_1 $	$\mathcal{P}_2$	C
1	1	1	1	1
1	0	1	1	0
0	1	0	1	1
0	0	0	0	0

The synthetic corpora approach examined in this paper could potentially help all these models to acquire better deductive reasoning ability.

### **B.** Limitations

The study has the following limitations:

- We examined only a kind of logical reasoning: deductive reasoning with a given set of facts. As stated in Section 8, we have other types of logical reasoning to be solved in the future.
- We examined only the first-order predicate logic system, while there are other logic systems useful for real-world reasoning to be tackled in the future, as stated in Section 8.

## C. Ethics and Social Impacts

The ultimate goal of the direction of this study is to make an artificial intelligence (AI) that reasons logically step by step. If AI can make a decision by showing logical steps one by one, it will be highly explainable and transparent to users. Furthermore, a user will be able to trace the error of AI. Thus, we think this study makes steps towards AI that will positively impact society.

## D. Formal Logic

### D.1. Definition of Validity of a Deduction Rule

A deduction rule is valid when for all truth value assignments, the conclusion is always true (=1) if all the premises are true (=1). This is exemplified in Table B.11a

A deduction rule is invalid when for some truth value assign-

ments, the conclusion is false (=0) even if all the premises are true (=1). This is exemplified in Table B.11b.

### D.2. Deductoin Rules used in Relevant Corpora

Figure B.4a shows the axioms of first-order predicate logic. Figure B.4b shows the deduction rules of implication type used in RuleTaker (Clark et al., 2021). For the deduction rules used in AACorpus, see Figure 1 in Betz et al. (2021).

## D.3. Examples of Multistep Deduction constructed from the Axioms

Figure B.5a shows the derivation of a deduction rule of implication type used in RuleTaker. Figure B.5b shows the derivation of the contraposition deduction rule used in AACorpus.

# E. Corpus Generation based on Formal Logic Deduction

We show the examples of FLD example in Figure C.6

Below, we show additional details of each module of **FLD**. Please refer to Figure 3 on intuitive understanding.

#### **E.1. Proof Tree Generation**

As stated in Section 3.1, the generator module generates a proof tree by forward- and backward random deduction, using the deduction rules specified by a user. A user can specify deduction rules via a template file as exemplified in Figure E.7.

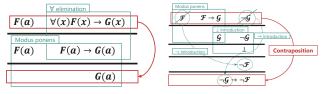
The forward random deduction is done as follows. The generator first chooses a deduction rule randomly and forms the initial tree where the root node is the conclusion of the chosen deduction rules and the child nodes are the premises of the chosen deduction rule. The generator next randomly chooses another deduction rule that can be "jointed" to the root note of the tree. A deduction rule can be jointed to the root node of a tree if one of the premises of that deduction rule can be identified with the root node. Then, the generator updates the tree by jointing this chosen deduction rule. The generator continues this step multiple times until the tree achieves the required depth.

The backward random deduction is done as follows. For each step, the generator randomly chooses a leaf node of the tree. Then, the generator randomly chooses a deduction rule that can be jointed to the leaf node. Here, a deduction rule can be jointed to the leaf node if the deduction rule's conclusion can be identified with the leaf node. Then, the generator updates the tree by jointing this chosen deduction rule. The generator continues this step multiple times until the complexity of branches achieves the required level.

$$\frac{\mathcal{F}}{\underbrace{(...)}_{g}} \rightarrow \text{introduction} \qquad \frac{\mathcal{F}}{g} \stackrel{(\mathcal{F} \rightarrow \mathcal{G})}{\text{g}} \rightarrow \text{elimination} \qquad \frac{\mathcal{F}}{g} \stackrel{(\mathcal{F} \land \mathcal{G})}{\text{g}} \wedge \text{introduction} \qquad \frac{\mathcal{F} \land \mathcal{G}}{g} \wedge \text{elimination} \qquad \frac{\mathcal{F} \land$$

(b) The deduction rules of the implication type used in RuleTaker (Clark et al., 2021).

Figure B.4: We show the deduction rules used in relevant corpora. For the "critical thinking" deduction rules used in AACorpus (Betz et al., 2021), please refer to Figure 1 in Betz et al. (2021).



(a) Derivation of a deduction rule of implication type used in RuleTaker (Clark et al., 2021).

(b) Derivation of contraposition deduction rule used in AACorpus (Betz et al., 2021).

Figure B.5: As examples of multistep deduction constructed from the axioms, we show the derivations of the deduction rules used in the previous studies.

### **E.2. Factual Distractor Generation**

We implemented three types of distractors. For a deductive nstance, we use the random mixture of these distractors. Below, we detail each type of distractor.

**Logical Distractor:** We construct a distractive formula the form of which is similar to a gold formula. For example, if a gold formula is  $(A \wedge B) \to C$ , then the following formula can be a distractor:  $(\neg A \wedge B) \to C$ . The aim of this type of distractor is to generate negative facts in a logic sense.

**Linguistic Distractor:** We construct a distractive sentence by randomly flipping a word in a gold sentence into another word. For example, if a gold sentence is "If it is not the fact that a sun rises, then (...)", then the following sentence can be a distractor: "If it is not the lion that a sun rises,

(...)" We considered grammatical constraints (e.g., part-of-speech) when flipping a word. Note that these distractors are made after the natural languages are assigned to gold formulas. The aim of this type of distractor is to generate negative facts in a linguistic sense.

**Negative Tree Distractor:** We create another proof tree irrelevant to the gold proof tree and use its leaf nodes as distractors. If a prover chooses these distractors as the starting point of the proof, then it will reach a conclusion that is irrelevant to the given hypothesis. Thus, this type of distractor measures the prover's look-ahead ability.

### E.3. Natural Language Assignment

We show an example of the natural language template file in Figure E.7.

When we assign natural language statements to each atomic component such as A, B, F, G, H, I, a, b, we used grammatical heuristics as exemplified as follows: (i) Atomic propositions like A and B are converted into complete-sentence statement like "[NOUN] is [ADJ]", "[NOUN] [VERB]" and "[NOUN] occurs". (ii) (Logical) predicates like F and G are converted into predicate phrases such as "[VERB]", "is [ADJ]" and "is [NOUN]". (iii) Constants like a are converted into entity-like phrases such as "[NOUN]".

```
sent1: as the factotum will tinge, the balbriggan is a Zinzendorf sent2: a factotum that is not an example of a show-stopper and also that does not exit is caused by the factotum not tingeing sent3: a factotum that is harmoniously an canary of a show-stopper and also that does not exit is caused by the factotum not tingeing sent4: it is wrong that, the factotum is attributable and also is sclerotic, because it is not true that the doubling will debunk massicot sent5: the massicot is not a kind of a lodger and also it is not illegitimate sent6: the balbriggan is incapable and also will promise strider sent7: a balbriggan that is incapable and also that does promise strider triggers the balbriggan that is not a Zinzendorf sent8: as a factotum will not host it is doubling sent9: a factotum that is attributable and also that is sclerotic is accelerated by the factotum that doubles sent10: if it is wrong that the massicot is Goethean, it is wrong that the doubling will debunk massicot sent11: because a factotum is sclerotic it will grass factotum sent12: it is not true that something does host, when it is incorrect that it does exit and it is not a kind of a show-stopper
                                                                                                                           : the factorum is attributable and also is sclerotic
 roof
                                                                                                                                                                                                                 let's assume that the factotum does tinge
the balbriggan is a kind of a Zinzendorf
it is not the fact that the balbriggan is a kind of a Zinzendorf
this is contradiction
                                                                                                                                                assump1:
int1:
int2:
int3:
   assump1
      sent6
int1
                                        sent7
int2
                                                                                                                                                                                                                this is Contradiction
it is incorrect that the factotum tinges
the factotum is a show-stopper and also is a kind of an exit
the factotum will exit and is an example of a show-stopper
when the factotum is an exit and is an example of the show-stopper, the factotum is not an example of a host
it is wrong that the factotum does host
a factotum that is not an example of a host gives rise to the factotum that is attributable and sclerotic
[assump1]
int4 s
                                                     int3
                                                                                                                                                               int4:
       int5
                                                                                                                                                               int6:
                                                                                                                                                              int7:
    sent12
int6
                                          sent9
int9
                                                                                                                                                               int9:
                                                                                                                                    hypothesis;
```

(a) An example where the proof by contradiction is used.

```
sent1: the monomer is an example of a power and it is an example of a abruption
 sent2: the grouper is a abruption sent3: the monomer is a kind of a abruption sent4: the monomer is an example of a bile and it is an example of a abruption
 sent5: the monomer is an example of a bite and it is an example of a abruption sent5: the monomer is inflectional sent7: the monomer is an example of a power and also it is axial sent8: the monomer is an example of a power and it is an example of a defilade sent9: the monomer is an increase and it is a kind of a KP
 sent10: the monomer will squeegee Haggard and will squeegee sardonyx sent11: the monomer is unquestionable and is inflectional
 sent12: the company does power
sent13: the monomer is calyculate and is inflectional
                                                 : the monomer is calyculate and also it will power
sent13
                                                                   int1:
                                                                                         the monomer is calyculate
                                                                                         the monomer is an example of a power
the monomer is a kind of a power that is calyculate
 sent1
                                                                   int2:
                 int2
  int1
                                                                   int3:
                                                        hypothesis;
 er: proved
```

(b) An example that can test the semantics of logical conjunction  $\wedge$ .

Figure C.6: Examples of FLD examples.

## F. Details of Experiments

### F.1. Corpora

As shown in Table 2, we generated multiple corpora to align conditions as closely as possible across the corpora being compared. Below, we detail each aspect.

**Preprocessing on RuleTaker:** We sub-sampled the examples so that the number of examples in the training split becomes 30k and that the answer label distribution (over "proved", "disproved", and "unknown") becomes uniform.

**Dataset Size:** All the corpora have the training split of 30k examples. FLD corpora have validation and test split of 1k examples. For RuleTaker, see (Tafjord et al., 2021).

**Label Distribution:** All the corpora have a uniform distribution over answer labels, i.e., over "proved", "disproved", and "unknown".

**Deduction Rules:** "Implication" deduction rules are shown in Figure B.4b. Complicated formula versions such as  $F_1(a) \wedge F_2(a) \rightarrow G(b)$  are also used. For "critical thinking", we used the ones listed in Figure 1 in Betz et al. (2021). For "axioms", we used the axioms of first-order predicate logic shown in Figure B.4a.

**Linguistic Diversity:** First, we detail the linguistic diversity of RuleTaker corpora. RT is classified into "less" since it uses only few templates for each logical statement. RT.PR is classified into "more" since it includes various

```
{
    "id": "implication_elim (modus_ponens)",
    "premises": [
        "{A} -> {B}",
        "id": "implication_intro",
        "premises": [
            "{A} + {B}"
],
    "conclusion": "{A} -> {B}"
},
    "conclusion": "{A} -> {B}"
},
    "id": "and_elim_6",
    "premises": [
        "({A} & {B})"
],
    "conclusion": "{A}"
},

"id": "and_elim_1",
    "premises": [
        "({A} & {B})"
],
    "conclusion": "{B}"
},

"id": "and_intro",
    "premises": [
        "(A)",
        "{B}"
],
    "conclusion": "({A} & {B})"
},
    "id": "or_intro_0",
    "premises": [
        "{A}",
        "feb"
],
    "id": "or_intro_0",
    "premises": [
        "{A}",
        "feb"
],
    "id": "or_intro_0",
    "premises": [
        "{A}",
        "id": "or_intro_0",
        "premises": [
        "{A}",
        "and_intro_0",
        "premises": [
        "[A]",
        "and_intro_0",
        "premises": [
        "[A]",
        "and_intro_0",
        "premises": [
        "[A]",
        "and_intro_0",
        "premises": [
        "[A]",
        "and_intro_0",
        "and_intro
```

Figure E.7: An example of the template rule file for deduction rules used by the proof tree generator.

paraphrases of logical statements collected from human annotators. RT.BE is classified into "more" since it is completely human-made and includes various paraphrases.

For FLD corpora, "less" means that we used only one natural language template for each logical statement and limited vocabulary (sized 100) for each POS. For "more", we used several (up to five) natural language templates for each logical statement, and a large vocabulary (sized 5000) for each POS. Note that since the templates can be nested such as:

$$(A \wedge B) \rightarrow C$$
: "If  $(A \wedge B)$ , then C.", " $(A \wedge B)$ , thus C."  $(A \wedge B)$ : "A, and B", "B and also A", . . .

Thus, the number of resulting patterns is combinatorially large.

**Formula Complexity:** For FLD corpora that have formula complexity "simple", we assign each compound formula such as  $\mathcal{F}$  and  $\mathcal{G}$  only a single atomic component as  $\mathcal{F}=A$  and  $\mathcal{G}=B$ . For "complex", we used compound formulas randomly constructed from atomic formulas with logical operators, such as  $\mathcal{F}=(A\wedge B), \ \mathcal{F}=\neg(A\vee \neg B),$  in addition to the "simple" formulas.

RuleTaker corpora use "complex" formula.

**Tree Depth:** The tree depth of "critical thinking" is limited to one because the critical thinking deduction rules have high-granularity, and thus cannot be easily combined to form multistep deductions.

**Tree Depth Distribution:** We have two types of tree depth distribution: skewed and uniform. The skewed distribution is biased toward lower depths. This distribution comes from the distribution of RT(D0-D3). The uniform distribution is uniform over the depths.

### F.2. Prover Training and Performance Measurement

We detail the prover training and performance measurements on the benchmarks. This experimental setting is basically the same as Yang et al. (2022). Thus, please refer to the study when necessary.

### F.2.1. THE PROVER MODEL

We added a slight modification to the original model for simplicity as follows. While the original model predicted an answer label (i.e., proved, disproved, or unknown) of a given example on the basis of the log-likelihood of the augmented proof sequences, we predict the answer label by forcing the prover to generate the label token ("\_\_PROVED\_\_", "\_\_DISPROVED\_\_" or "\_\_UNKNOWN\_\_") at the end of proof sequences.

# F.2.2. Few-shot Transfer to Synthetic Deduction Corpora

We first train a prover on the training split of a corpus in Table 2. Then, we train the prover on a training split of another corpus in few-shot, and after that, we measured its performance on the test split. We used validation split for tuning hyperparameters. We adopted T5-base for prover LM for computational efficiency. Table F.12 shows the hyperparameters. For the "fully-fine-tuning" setting used in Section 5 and Appendix G, we trained a prover using all the dataset examples for 20k steps.

We run the experiments for one seed for computational reasons. Training on a source corpus takes about ten hours on a single NVIDIA A100 (48GB) GPU. Training on a target corpus takes a few hours on the same GPU.

### F.2.3. TRANSFER TO ENTAILMENTBANK

We first train a prover on the training split of a corpus from Table 2. Then, we train the prover on the training split of each EntailmentBank corpus. We adopted T5-large for prover LM following Yang et al. (2022). The hyperparameters are basically the same as Yang et al. (2022). Table F.13

```
"(x): {A}x -> {B}x": [
"csphrase::if_conjunction>> <sentence::{A}something>>, <sentence::{B}it>>",
"<sentence::{A}something>>, <sphrase::thus_conjunction>> <sentence::{B}it>>",
"<sentence::{B}something>>, <sphrase::thus_conjunction>> <sentence::{B}it>>",
"<sentence::{B}something>>, <sphrase::fe_conjunction>> <sentence::{A}it>>",
"<sclause::noun::{A}something.set>> <sphrase::fe_conjunction>> <sentence::{A}it>>",
"<sphrase::every_prefix.sentence_intro>>, <sphrase::if_conjunction>> <sentence::{B}it>>",
"<xphrase::fe_conjunction>> <sentence::{A}something>>, <sphrase::thus_conjunction>> <sentence::{B}it>>",
"<sentence::{A}something>>, <sphrase::thus_conjunction>> <sentence::-{A}it>>",
"<clause::noun::-{A}something.set>> <sphrase::fe_conjunction>> <sentence::-{A}it>>",
"<xphrase::every_prefix.sentence_intro>>, <sphrase::fe_conjunction>> <sentence::-{A}it>>",
"<xphrase::every_prefix.sentence_intro>>, <sphrase::fe_conjunction>> <sentence::-{B}it>>",
"<xsentence::{B}something>>, <sphrase::thus_conjunction>> <sentence::-{B}it>>",
"<xsentence::{B}something>>, <sphrase::funcion>> <sentence::{A}it>>",
"<xcentence::{A}something>>, <sphrase::fe_conjunction>> <sentence::{A}it>>",
"<xcentence::{A}something>, <sphrase::fe_conjunction>> <sentence::{B}it>>",
"<xcentence::{A}something>, <sphrase::fe_conjunction>> <sentence::{A}it>>",
"<xcentence::{A}something>>, <sphrase::fe_conjunction>> <sentence::{A}it>>",
"<xcentence::{A}something>>, <sphrase::thus_conjunction>> <sentence::{A}it>>",
"<xcentence::{A}something>>, <sphrase::thus_conjunction>> <sentence::{A}it>>",
"<xcentence::{A}it>>",
"<xcentence::{A}something>>, <sphrase::thus_conjunction>> <sentence::{A}it>>",
"<xcentence::{A}it>>",
"<xcen
```

```
"if_conjunction": [
"if",
"when",
"as",
"because",
"since"
],
"thus_conjunction": [
"thus",
"therefore",
"so that"
],
"not_prefix.sentence_intro": [
"it is not <<phrase::correct_adj>> that",
"it is <<phrase::incorrect_adj>> that"
],
"correct_adj": [
"the fact",
"true"
],
"incorrect_adj": [
"wrong",
"incorrect",
"false"
],
"occur": [
"occurs",
"happens",
"takes its rise"
],
```

(b) The final natural language templates of logical phrases.

(a) For each formula, we have several templates. A template can be nested (redirected to other templates), as shown.

Figure E.8: An example the template file of the natural language assigner

Table F.12: The hyperparameters of prover training in the transfer experiments among deduction corpora. See (Yang et al., 2022) or the code for other parameters.

	Source corpus	Target corpus
transformer model	T5-base	T5-base
# dataset examples	30000	300
steps	20000	2000
learning rate	1e-4	1e-4
learning rate scheduler	AdamW	AdamW
warmup steps	1000	500
batch size	64	64
gradient clipping	0.5	0.5

transfer experiments from deduction corpora to Entailment-Bank. See (Yang et al., 2022) or the code for other parameters.

Table F.13: The hyperparameters of prover training in the

	Source synthetic corpus	Traget EB corpus
transformer model	T5-large	T5-large
# dataset examples	30000	1313
steps	10000	10000
learning rate	1e-4	5e-5(task1), 2.5e-5(task2)
learning rate scheduler	AdamW	AdamW
warmup steps	1000	1000(task1), 3000(task2)
batch size	64	64
gradient clipping	0.5	0.5

shows the hyperparameters.

For training the verifier, we used exactly the same setting as Yang et al. (2022).

For the EB scorer, we used the same version as Yang et al. (2022), that is, version v3 that was released on May 28,  $2022^{7}$ .

We run the experiments for six seeds. Training on a source synthetic deduction corpus takes about one day on a single NVIDIA A100 (48GB) GPU. Training on a target EB corpus

takes about one day on the same GPU.

### F.3. License of used Datasets.

All the datasets used in this paper are publicly available: RuleTaker (Clark et al., 2021; Tafjord et al., 2021), EntailmentBank (Mishra et al., 2022) and FLD (will be publicly available).

<sup>7</sup>https://github.com/allenai/entailment\_ bank

## G. How Well do LMs Solve Logic?

Table G.14: Proof accuracy of a prover fully fine-tuned on each corpus using all the examples.

Skewed depth distrib.		Uniform depth distrib.				
RT	RT.PR	sFLD-impl	FLD-impl.0	FLD.2	FLD.3 (FLD)	FLD.4 ( <b>FLD</b> ★)
92.4	93.9	82.2	74.6	66.8	66.4	37.7

We analyze the challengingness of each corpus in detail by using Table G.14.

First, we look into the results on skewed depth distribution corpora. As seen, sFLD-impl is more challenging than Rule-Taker corpora. Since the corpus design in relevant aspects is aligned between RuleTaker and sFLD-impl as in Table 2, the difference should come from the other implementation details. For example, the distractors of FLD are designed to be easily confused with positive facts (Section 3.2), and natural language assignments are extremely diverse due to the random statement generation (Section 3.3).

FLD-impl.0 is more challenging than sFLD-impl even though they use the same deduction rules. This should be because FLD-impl.0 has "uniform" tree depth distribution and thus includes a higher depth tree than sFLD-impl do (Appendix F.1).

The reason that FLD.2 is more challenging than FLD-impl.0 should be as follows. First, since a proof tree is constructed from the combination of deduction rules chosen at each level of the tree, the number of possible proof tree patterns can be estimated (very roughly) as  $\mathcal{O}(\mathcal{A}^d)$ , where  $\mathcal{A}$  is the number of deduction rule choices at each level and d is the depth of the proof tree. Next, while FLD-impl.0 uses only a few deduction rules ( $\mathcal{A}=2$ ) of implication type shown in Figure B.4b, FLD.2 uses various deduction rules ( $\mathcal{A}\sim10$ ) of the axioms shown in Figure B.4a. Thus, FLD.2 includes exponentially more diverse patterns of proof trees than RuleTaker. This makes FLD.2 more challenging than FLD-impl.0 .

FLD.3 is the linguistically diverse version of FLD.2. The challengingness of FLD.3 remains almost the same as that of FLD.2 provably because LMs can solve the linguistic aspects such as paraphrasing, as discussed in Section 7.2.

FLD.4 is the higher-depth (d up to 8) version of FLD.3. This corpus is the most challenging provably due to the exponentially combinatorially more diverse patterns proof trees coming from  $\mathcal{O}(\mathcal{A}^d)$ .

# G.1. Answer Accuracies on Transfer Experiments among Deduction Corpora

Below, we show the results of transfer among synthetic corpora measured by the other metric of answer accuracy.

Table G.15: Answer accuracy of a prover fully fine-tuned using all the dataset examples on each corpus.

RT	RT.PR	sFLD-impl	FLD-impl.0	FLD.2	FLD.3	FLD.4
95.2	95.8	96.1	94.9	88.3	87.7	68.1

Table G.16: Few-shot answer accuracies of provers transferred among corpora that differ in deduction rules.

	T5	RT	RT.PR	sFLD-impl	sFLD-crit	sFLD
RT	80.7	95.2	93.7	83.6	83.2	84.8
RT.PR	78.4	93.0	95.8	82.5	79.7	82.4
RT.BE	56.2	88.3	88.2	75.2	79.4	85.0
FLD (RT)	78.3	83.1	83.3	96.1	86.0	95.3
FLD (AA)	84.9	85.5	84.5	91.7	95.2	95.1
FLD (axiom)	79.0	76.9	76.9	87.3	85.7	92.9
avg.	76.3	87.0	87.1	86.1	84.9	89.3

Table G.17: The depth-wise answer accuracies of the provers.

(a) Target corpus is FLD-impl.1. (b) Target corpus is FLD.4.

			_	Source	corpus	
		Source	corpus	T5	FLD.3	FLD.4
	T5	FLD-impl.0	FLD-impl. i	75.0	100.0	100.0
0	50.0	91.7	87.5	82.4	98.6	95.9
1	96.8	98.4	96.8	77.3	89.8	91.5
2	88.0	97.0	94.0	78.9	84.2	78.9
3	90.0	93.8	96.2	76.7	79.5	71.2
4	88.9	93.3	90.0	70.4	65.7	60.6
5	72.6	90.5	95.2	70.6	63.5	51.8
6	78.7	89.3	98.7	77.1	52.9	47.1
7	84.0	90.6	91.5	71.1	47.0	45.2
8	82.0	90.2	91.8	75.5	75.7	71.4
avg.	81.2	92.8	93.5			

Table G.18: Few-shot answer accuracies of provers transferred among corpora that differ in the diversity of linguistic expressions.

	T5	RT	RT.PR	FLD.2	FLD.3
RT	80.7	95.2	93.7	85.2	83.9
RT.PR	78.4	93.0	95.8	80.8	82.0
FLD.2	72.1	72.1	71.3	88.3	86.9
FLD.3	68.2	68.0	67.7	86.7	87.7
avg.	74.9	82.1	82.1	85.3	85.1

Table G.19: Few-shot answer accuracies of provers transferred among corpora that differ in the complexity of formulas

	T5	FLD.1	FLD.2
FLD.1	77.9	96.5	91.6
FLD.2	72.1	77.8	88.3

Table G.20: Few-shot answer accuracies of provers transferred among corpora that differ in the number of distractors.

	T5	FLD.0	FLD.2
FLD.0	77.7	95.8	93.4
FLD.2	72.1	83.1	88.3

### G.2. Results of Other Metrics on EntailmentBank

We show the results of other metrics on EntailmentBank in Tables G.21 to G.23.

## G.3. Case Study on EntailmentBank

Table G.24 shows some cases where the error of the baseline prover (T5) is fixed by the training on a deduction corpus (**FLD.D5**).

As seen from "T5 error fixed by FLD.D5" column, typical error of T5 is such as follows: (i) T5 misses some of the premises required to derive the required conclusion, or, simply choose wrong premises. (ii) T5 overclaims, that is, included in the generated conclusion such information that does not logically follow from the chosen premises. It is also suggested that T5 does not understand the rules of logical operators such as negation  $\neg$  and conjunction  $\land$ .

In contrast, the prover trained on **FLD.D5** captured the fundamentals of deduction rules better than the baseline: (i) it chose correct premises necessary and sufficient to derive the next conclusion, (ii) it included in a conclusion only such information that logically follows from the chosen premises, and (iii) it correctly used the rules of logical operators.

### H. About the Version 2 of FLD

The current official release<sup>1</sup> of FLD corpora is version 2. In version 2, we fixed a proof inconsistency issue of version 1; specifically, the version 1 corpus contained some examples with contradicting proofs, such as "proved" versus "disproved." Due to this issue, the prover's performance had been underestimated. Fixing this issue, the performance of the provers on version 2 differs from version 1, as follows:

Table H.25: Proof accuracy of a prover fully fine-tuned using all the dataset examples on each corpus. Each corpus is version 2.

FLD.v2			
FLD	FLD∗		
75.8	44.4		

Table H.26: Answer accuracy of a prover fully fine-tuned using all the dataset examples on each corpus. Each corpus is version 2.

FLD.v2			
FLD	FLD∗		
91.6	72.2		

Table G.21: The results of all the metrics on EntailmentBank Task1. For the details of these metrics, refer to Dalvi et al. (2021).

Task	Leaves		Steps		Intermediates		Overall
	F1	AllCorrect	F1	AllCorrect	F1	AllCorrect	AllCorrect
T5	98.2	90.6	53.8	40.3	72.1	39.7	36.8
RT.D5	98.2	88.6	56.0	42.7	72.8	41.6	39.4
FLD.D5	99.0	92.7	55.5	42.2	73.4	41.3	39.2

Table G.22: The results of all the metrics on EntailmentBank Task2. For the details of these metrics, refer to Dalvi et al. (2021).

Task	Leaves		Steps		Intermediates		Overall
	F1	AllCorrect	F1	AllCorrect	F1	AllCorrect	AllCorrect
T5	88.1	52.1	45.9	33.2	68.6	36.2	31.2
RT.D5	88.6	53.1	45.6	32.7	68.4	36.1	32.0
FLD.D5	88.4	53.6	45.6	33.8	67.9	36.1	32.6

Table G.23: The results of all the metrics on EntailmentBank Task3. For the details of these metrics, refer to Dalvi et al. (2021).

Task	Leaves		Steps		Intermediates		Overall
	F1	AllCorrect	F1	AllCorrect	F1	AllCorrect	AllCorrect
T5	43.7	8.2	10.7	6.5	42.2	16.8	6.2
RT.D5	43.1	10.0	13.1	8.2	41.7	17.3	8.2
FLD.D5	43.6	9.7	12.1	8.3	43.0	20.1	8.3

Table G.24: Cases where baseline (T5) error on a proof step is fixed by the training on **FLD.D5**. A proof step is composed of a set of chosen premises and a derived conclusion. A derived conclusion is either a sequence generated by the prover model when the step is an intermediate step of the proof or is (fixed to) the hypothesis given to the model when the step is the final step of the proof (marked as [hypothesis]). Thus, in the final step, the task of the prover model is just to choose the right premises that can derive the hypothesis.

	chosen premises	derived conclusion	T5 error fixed by <b>FLD.D5</b>	
FLD.D5	force causes the speed of an object to increase / to decrease     an increase is a kind of change     a decrease is a kind of change	force causes the speed of an object to change [hypothesis]	The chosen premises are not sufficient, possibly happened because T5 do not understand the rule of conjunction (\(^\)).	
T5	force causes the speed of an object to increase / to decrease     an increase is a kind of change	force causes the speed of an object to change [hypothesis]		
FLD.D5	1. the milky way is made of stars 2. light year can be used to measure the distance between stars  light year can be used to measure the distance between the stars in milky way		T5 overclaimed, that is, included in the conclusion such information	
T5	1. the milky way is made of stars	the distance between the stars in milky way is light years	that does not logically follow from the chosen premises. (a.k.a "hallucination")	
FLD.D5	1. fossils are formed by the remains of living things 2. rock is a kind of nonliving thing rocks cannot form fossils [hypothesis]		T5 missed a premise to choose wrong one, possibly because T5 do not understand	
T5	fossils are formed by the remains of living things     cannot means not be able to	rocks cannot form fossils [hypothesis]	the semantics of negation (¬) at "nonliving".	
FLD.D5	the first quarter phase of the moon occurs after the new moon     different moon phase occurs approximately once per week	the first quarter phase of the moon will occur one week after the new moon [hypothesis]	T5 chose wrong premises.	
T5	1. the first quarter phase of the moon occurs after the new moon 2. old is the opposite of new	the first quarter phase of the moon will occur one week after the new moon [hypothesis]		
FLD.D5	cutting down trees has a negative impact on an environment     decreasing something negative has a positive impact on a thing	decreasing the amount of trees cut has a positive impact on the environment	T5 generation was merely a repetition of a premise (a.k.a "repetition")	
T5	cutting down trees has a negative impact on an environment     creating paper requires cutting down trees	cutting down trees has a negative impact on the environment		
FLD.D5	the heart is situated in the chest     chest and abdomen are different location on a body	the abdomen does not contain the heart [hypothesis]	T5 chose wrong premises.	
T5	the heart is situated in the chest     if something does not contain something else then that something lacks that something else	the abdomen does not contain the heart [hypothesis]		