Proceedings of the 23rd International Conference on Principles of Knowledge Representation and Reasoning
KR meets Machine Learning and Explanation

Hybrid Models for Natural Language Reasoning: The Case of Syllogistic Logic

Manuel Vargas Guzmán[1,2], **Jakub Szymanik**[3], **Maciej Malicki**[1]
[1]University of Warsaw
[2]Institute of Fundamental Technological Research, Polish Academy of Sciences
[3]University of Trento
m.vargas-guzman@uw.edu.pl, jakub.szymanik@gmail.com, mmalicki@mimuw.edu.pl

Abstract

Despite the remarkable progress in neural models, their ability to generalize—a cornerstone for applications like logical reasoning—remains a critical challenge. We delineate two fundamental aspects of this ability: compositionality, the capacity to abstract atomic logical rules underlying complex inferences, and recursiveness, the aptitude to build intricate representations through iterative application of inference rules. In the literature, these two aspects are often confounded together under the umbrella term of generalization. To sharpen this distinction, we investigated the logical generalization capabilities of pre-trained large language models (LLMs) using the syllogistic fragment as a benchmark for natural language reasoning. We extend classical Aristotelian syllogistic forms to build more complex structures, providing a foundational yet expressive subset of formal logic that supports controlled evaluation of essential reasoning abilities. Our findings reflect this non-trivial benchmark: while LLMs demonstrate reasonable proficiency in recursiveness, they struggle with compositionality. This disparity, however, is not uniform, as a more detailed analysis reveals variability in generalization performance across individual syllogistic types, ranging from near-perfect to significantly lower accuracy. To overcome these limitations and establish a reliable logical prover, we propose a hybrid architecture integrating symbolic reasoning with neural computation. This synergistic interaction enables robust and efficient inference—neural components accelerate processing, while symbolic reasoning ensures completeness. Our experiments show that high efficiency is preserved even with relatively small neural components. As part of our proposed methodology, this analysis gives a rationale and highlights the potential of hybrid models to effectively address key generalization barriers in neural reasoning systems.

1 Introduction

Neural models have achieved substantial advancements at an accelerated pace in recent years. However, they continue to face challenges in generalizing—a capability that is crucial for tasks such as logical deduction. While they excel at pattern recognition, these models often struggle with the systematicity and robustness required for sound reasoning (Marcus, 2018; Lake et al., 2017; Huang and Chang, 2023; Mondorf and Plank, 2024). This is particularly evident in their limited capacity to generalize beyond the training data, especially in tasks that demand a deep understanding of compositional structures (Hupkes et al., 2023).

In this work, we focus on two fundamental and complementary aspects of generalization in the context of logical reasoning: compositionality and recursiveness. *Compositionality* is the ability to recognize that the logical rules governing simple inferences remain valid when embedded within more complex structures. While the Principle of Compositionality (Janssen and Partee, 1997) is traditionally framed as a "bottom-up" process—where the meaning of a complex expression is determined by the meanings of its constituent parts, we diverge from this standard usage to emphasize an analytical, "top-down" perspective. Under our definition, a system is compositional if it can abstract atomic logical rules from complex structures and apply them to its simpler components. For example, if a model can derive the inference "if all a are b, all b are c, and all c are d, then all a are d," it should also be able to derive the constituent inference "if all a are b and all b are c, then all a are c." *Recursiveness*, on the other hand, is the ability to construct complex representations through the iterative application of a finite set of rules. While recursiveness is often viewed as the constructive mechanism of compositionality, we treat it as a distinct property to isolate a model's capacity for length-expansion. For instance, a recursive model can extend the transitive inference "if all a are b and all b are c, then all a are c" to a longer chain: "if all a are b, all b are c, and all c are d, then all a are d." A system that is merely recursive might produce longer chains by mimicking surface-level patterns, but a truly compositional system demonstrates an understanding of the argument's structure by successfully reasoning about its sub-parts. It is therefore possible for a model to be recursive—processing arbitrarily long chains—without being compositional, meaning it lacks an understanding of how individual links in the chain combine. This lack of analytical compositionality remains a key limitation of current neural models (Vargas Guzmán, Szymanik, and Malicki, 2024; Lake and Baroni, 2023; Bertolazzi et al., 2026).

To investigate this issue, we examine the logical generalization capabilities of large pre-trained language models (LLMs) using syllogistic logic—a well-defined, yet non-trivial, fragment of natural language that captures a fundamental aspect of human reasoning. Syllogistic logic was chosen as a clearly tractable baseline for compositional and recursive generalization, as logics that are too

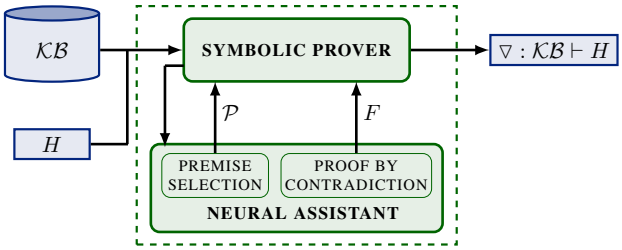


Figure 1: Overview of the hybrid architecture. Input: a knowledge base $\mathcal{KB}$ and a hypothesis H. Hybrid model: the neural models assist the symbolic prover by providing a subset $\mathcal{P} \subseteq \mathcal{KB}$ such that $\mathcal{P} \vdash H$, and a refutation formula F such that $\mathcal{KB} \cup \{\overline{H}\} \vdash F$ and $\mathcal{KB} \vdash \overline{F}$. Output: a proof $\triangledown$ of H from $\mathcal{KB}$.

expressive—such as full first-order logic—are computationally intractable. We fine-tuned LLMs on two distinct reasoning tasks to support the construction of direct and indirect formal proofs. Specifically, for a given knowledge base $\mathcal{KB}$ and a hypothesis H, the first task is *premise selection*, i.e., predicting a minimal subset $\mathcal{P} \subseteq \mathcal{KB}$ such that $\mathcal{P} \vdash H$; the second task is *proof by contradiction*, i.e., predicting a formula F such that $\mathcal{KB} \cup \{\overline{H}\} \vdash F$ and $\mathcal{KB} \vdash \overline{F}$. These tasks were designed to probe different facets of logical generalization, with premise selection requiring an understanding of the relationships between statements and proof by contradiction testing the ability to reason about counterfactuals and derive logical consequences. To ensure a rigorous evaluation, we trained and tested on multiple knowledge bases generated from controlled synthetic data, which incorporates pseudowords to avoid content bias (Bertolazzi, Gatt, and Bernardi, 2024). Our experiments reveal a significant disparity: while LLMs demonstrate reasonable proficiency in recursive reasoning, they struggle with compositional generalization. Specifically, when trained on simpler inferences, LLMs can recognize analogous simple inferences across different knowledge bases and generalize to a certain extent to more complex inference patterns. However, models trained exclusively on complex inferences exhibit a substantial performance drop when required to identify the underlying simpler components. Importantly, we observe systematic variability in performance across different reasoning structures, indicating that generalization is type-dependent. Models generalize certain forms consistently, yet struggle with others. That these phenomena arise already within syllogistic logic is theoretically significant. Despite its simplicity, this fragment reveals a clear structural boundary between inference types that LLMs can generalize and those they cannot.

To address this limitation, we propose a new research program consisting of two elements. On the theoretical side, we aim to understanding how different reasoning building blocks interact with deep-learning model performance on generalization tasks. On the practical side, we develop a hybrid architecture (see Figure 1) that integrates the pattern-matching strengths of neural networks with the formal rigor and completeness of symbolic reasoning. In this framework, the neural component serves as an auxiliary to the symbolic prover, efficiently providing candidate premises and formulas to guide the derivation process. Furthermore, to evaluate the impact of the assistant on the symbolic prover in terms of the number of computational steps required during proof search, we implemented a baseline non-deterministic prover. This synergistic approach aims to overcome the limitations of purely neural approaches by enforcing logical consistency and enabling systematic generalization.

The key contributions of this work are: (1) A rigorous empirical demonstration that, despite their recursive capabilities, LLMs lack true compositionality, a crucial requirement for robust logical reasoning. We emphasize the importance of distinguishing between these two properties in evaluations of neural reasoning systems. (2) A theoretical finding that generalization in neural models is constrained by logical structure. This highlights that certain inference types can be effectively learned while others pose difficulties, providing a systematic analysis for understanding the limits of neural reasoning. (3) A hybrid approach that leverages neural networks for efficient inference while relying on symbolic reasoning to guarantee logical completeness and correctness. We evaluate the robustness of the hybrid model, focusing on how neural assistance affects symbolic proof search across different model sizes and configurations.

We present our hybrid framework in three parts. The first part establishes the symbolic foundation of the approach: we introduce the underlying syllogistic logic (Section 2) and a non-deterministic algorithm that implements a prover (Section 3). The second part focuses on the connectionist setting, where we train neural models to approximate this reasoning process. We describe the synthetic dataset and fine-tuning setup (Section 4), and then analyze how these models generalize beyond their training distribution, relating performance to structural properties of syllogistic proofs (Section 5). The third part brings both perspectives together by examining how the symbolic and neural components interact within a unified experimental setting (Section 6).

2 A Syllogistic Proof System

We adopt a syllogistic proof system based on Smiley (1973) as a foundation for implementing hybrid reasoning models. The system comprises four formula types: Aab, Eab, Iab, and Oab, corresponding to the natural-language forms "All a are b", "No a are b", "Some a are b", and "Some a are not b." This framework extends the classical two-premise syllogism by allowing derivations from arbitrarily many premises. In particular, A-formulas are treated as chains representing transitive inclusion among terms. These chains provide the structural backbone of syllogistic reasoning, enabling its compositional and recursive analysis.

2.1 Syntax and Semantics

We formally define the syntax and semantics of a language with four *quantifier* symbols $\mathcal{Q} = \{A, E, I, O\}$ and an infinite set of *term* symbols $\mathcal{X} = \{a, b, c, \ldots\}$ denoted by lowercase letters (sometimes with subscripts). *Well-formed formulas* over this language are built as Aab, Eab, Iab, or Oab. An *A-chain*, denoted as $Aa-b$, represents either

the formula Aab or the sequence of two or more formulas $Aac_1, Ac_1c_2, \ldots, Ac_{n-1}c_n, Ac_nb$ (for $n \geq 1$). In what follows, when we refer to a formula we mean a well-formed formula. Moreover, we use capital letters (e.g., F or H) to denote formulas, and capital calligraphic letters (e.g., $\mathcal{KB}$, $\mathcal{F}$ or $\mathcal{P}$) to denote sets of formulas, unless stated differently.

The meaning of syllogistic formulas can be defined using set-theoretic relationships. Let us interpret terms a and b as non-empty subsets of an underlying universe M; then Aab is true *iff* $a \subseteq b$; Eab is true *iff* $a \cap b = \emptyset$; Iab is true *iff* $a \cap b \neq \emptyset$; and Oab is true *iff* $a \not\subseteq b$. A set of syllogistic formulas $\mathcal{F}$ is *consistent* if there exists an interpretation (i.e., a universe M and an assignment of subsets of M to terms) under which every $F \in \mathcal{F}$ is true; otherwise it is *contradictory*.

Note that Aab and Oab are contradictory, and the same about Iab and Eab. We denote the negation of a formula F as $\overline{F}$, i.e., $\overline{Aab} = Oab$, $\overline{Oab} = Aab$, $\overline{Iab} = Eab$, and $\overline{Eab} = Iab$. Last but not least, I- and E-formulas are symmetrical. That is, Iab and Iba have the same meaning, and so do Eab and Eba.

2.2 Inference and Proofs

We define a syllogism as an inference from a set of premises, referred to as a knowledge base, to a conclusion.

Definition 1 (Proof)**.** The following is a mutually recursive definition to characterize formal proofs from a set of formulas using tree notation. A *proof* $\triangledown$ of a hypothesis H from a set of premises $\mathcal{F}$ is one of the following three types:

(i) *Trivial proof*: for every $H \in \mathcal{F}$, the following tree is a proof of H from $\mathcal{F}$

$$\frac{}{H}\ \text{(i)}$$

(ii) *Rule-based proofs*: for $\triangledown'$ and $\triangledown''$ that are proofs from $\mathcal{F}$, the following trees are proofs from $\mathcal{F}$

$$\frac{\begin{array}{cc}\triangledown' & \triangledown'' \\ Aab & Abc\end{array}}{Aac}\ \text{(r1)} \qquad \frac{\begin{array}{cc}\triangledown' & \triangledown'' \\ Aab & Ebc\end{array}}{Eac}\ \text{(r2)}$$

$$\frac{\begin{array}{c}\triangledown' \\ Eba\end{array}}{Eab}\ \text{(r3)} \qquad \frac{\begin{array}{c}\triangledown' \\ Aba\end{array}}{Iab}\ \text{(r4)}$$

(iii) *Proof by contradiction*: for $\triangledown'$ that is a proof from $\mathcal{F} \cup \{\overline{H}\}$ and $\triangledown''$ that is a proof from $\mathcal{F}$, the following tree is a proof from $\mathcal{F}$

$$\frac{\begin{array}{cc}\triangledown' & \triangledown'' \\ F & \overline{F}\end{array}}{H}\ \text{(iii)}$$

Definition 2 (Inference)**.** Let $\mathcal{F}$ be a set of formulas (*premises*) and let F be a formula (*conclusion*). We write $\mathcal{F} \vdash F$, to denote that F is provable (or derivable) from $\mathcal{F}$, i.e., there exists a proof of F from $\mathcal{F}$.

τ	**Syllogism**
1	$\{Aa{-}b, Ac{-}d, Oad\} \vdash Obc$
2	$\{Aa{-}b\} \vdash Aab$
3	$\{Aa{-}b, Ac{-}d, Aa{-}e, Ede\} \vdash Obc$
4	$\{Aa{-}b, Aa{-}c\} \vdash Ibc$
5	$\{Aa{-}b, Ac{-}d, Ae{-}f, Iae, Edf\} \vdash Obc$
6	$\{Aa{-}b, Ac{-}d, Ebd\} \vdash Eac$
7	$\{Aa{-}b, Ac{-}d, Iac\} \vdash Ibd$

Table 1: Types of syllogistic inferences

Soundness and Completeness The deductive system presented above is based on the framework developed by Smiley (1973). In this system, a formula is provable if and only if it is true in all interpretations (as established in Theorems 3 and in 4). It therefore follows that the system is both sound and complete.

Definition 3 (Minimal Inference)**.** An inference $\mathcal{F} \vdash F$ is *minimal* if for no proper subset $\mathcal{F}' \subsetneq \mathcal{F}$, it is the case that $\mathcal{F}' \vdash F$.

Minimal inferences are essential to design a connectionist prover assistant, as they use only the necessary premises to derive a conclusion. Table 1 depicts all types of minimal syllogistic inferences, denoted by τ, as outlined in Vargas Guzmán, Szymanik, and Malicki (2024).

3 Symbolic Component

We implemented a basic automated syllogistic prover that takes as input a knowledge base $\mathcal{KB}$ and a hypothesis H, i.e., a proposed conclusion to be derived from $\mathcal{KB}$. The overall procedure is described in Algorithm 1. The prover first attempts to construct a proof using types (i) and (ii) via the DRV function. If this attempt fails, it proceeds to use proof by contradiction (type (iii)) through the PBC function. Successful partial derivations are stored in a set Δ as triples that correspond to proof steps: a set of premises $\mathcal{F}$, a derived formula F, and the proof type by which the derivation is obtained. If H is valid, the prover returns its corresponding derivation steps (line 3). Otherwise, it exhaustively explores all possible derivations before concluding that H is invalid.

3.1 Derivation of Formulas

The central component of the prover is the recursive function DRV, described in Algorithm 2. The function takes as input

Algorithm 1 Syllogistic Prover

Input: A hypothesis H and a knowledge base $\mathcal{KB}$
Output: A proof $\triangledown$ of H from $\mathcal{KB}$ (if H is valid)
1: $\Delta \leftarrow \emptyset$
2: **if** DRV$(H, \mathcal{KB})$ **or** PBC$(H, \mathcal{KB})$ **then**
3: $\quad \triangledown \leftarrow$ GET_STEPS(H, Δ)
4: $\quad$ **return** $\triangledown$
5: **else**
6: $\quad$ **return** `False`
7: **end if**

Algorithm 2 Derive (DRV)

Input: A hypothesis H and a knowledge base $\mathcal{KB}$
Output: `True` iff H is provable

1: **if** $H \in \mathcal{KB}$ **then**
2: $\quad\Delta \leftarrow \Delta \cup \{(\emptyset, H, \text{(i)})\}$
3: $\quad$**return** `True`
4: **else if** IS_DERIVABLE($\mathcal{F}, H$) **then**
5: $\quad\Delta \leftarrow \Delta \cup \{(\mathcal{F}, H, \text{(ii)})\}$
6: $\quad$**for all** $F \in \mathcal{F}$ **do**
7: $\quad\quad$**return** DRV($F, \mathcal{KB}$)
8: $\quad$**end for**
9: **else**
10: $\quad$**return** `False`
11: **end if**

the knowledge base $\mathcal{KB}$ and the hypothesis H, and initially checks whether proof of type (i) can be directly applied. If the base case is not satisfied, the function attempts to derive H by non-deterministically searching for a set of formulas $\mathcal{F}$ such that H follows from $\mathcal{F}$ (line 4) by using rules of inference, i.e., proofs of type (ii). Then the process is applied recursively to each $F \in \mathcal{F}$, continuing until the base case is met or no further applicable rules and formula combinations are available. To prevent redundant computations and potential infinite loops, the algorithm is optimized by storing partial proofs (lines 2 and 5) whenever the base case is reached. These stored derivations are reused if the same inputs are encountered again. Additionally, the system tracks failed derivation attempts to avoid repeating them in subsequent searches. Nevertheless, this search process remains computationally demanding, particularly when constructing *A-chains*, where the algorithm conducts a non-deterministic search over formulas of the form Aab, generated using terms drawn from the knowledge base. For each step, there are up to $n(n-1)$ candidates, where n denotes the number of distinct terms in the knowledge base, with each choice leading to further alternatives and causing the search space to grow rapidly.

3.2 Proof by Contradiction

The final component of the prover, denoted PBC, is specified in Algorithm 3. This function gets the same inputs as DRV, and it aims to prove the hypothesis H by contradiction by finding a formula F such that $\mathcal{KB} \cup \{\overline{H}\} \vdash F$ and $\mathcal{KB} \vdash \overline{F}$. The algorithm begins by generating all possible contradictory formula pairs $(F, \overline{F})$, where F ranges over all formulas obtained by applying quantifiers from $\mathcal{Q}$ to pairs of terms drawn from $\mathcal{X}$, with $\mathcal{X}$ restricted to the terms occurring in $\mathcal{KB}$. It then iterates through these pairs in search of a contradiction. If such proof is found, it is stored; otherwise, the process continues until all pairs have been exhausted. The search for candidate pairs is performed in a non-deterministic manner, which can result in significant computational cost, as the algorithm calls the DRV function for each formula within every potential pair.

Algorithm 3 Proof by Contradiction (PBC)

Input: A hypothesis H and a knowledge base $\mathcal{KB}$
Output: `True` iff H is provable by contradiction

1: $\mathcal{C} \leftarrow$ ALL_PAIRS($\mathcal{Q}, \mathcal{X}$)
2: **for all** $(F, \overline{F}) \in \mathcal{C}$ **do**
3: $\quad$**if** DRV($\overline{F}, \mathcal{KB}$) **and** DRV($F, \mathcal{KB} \cup \{\overline{H}\}$) **then**
4: $\quad\quad\Delta \leftarrow \Delta \cup \{(\{F, \overline{F}\}, H, \text{(iii)})\}$
5: $\quad\quad$**return** `True`
6: $\quad$**end if**
7: **end for**
8: **return** `False`

4 Connectionist Component

Our hybrid models integrate two distinct connectionist components designed to support the prover: one assists in selecting the necessary premises required to derive the hypothesis, while the other facilitates the identification of a suitable formula for proof by contradiction. To develop these models, we generated synthetic training data and fine-tuned pre-trained large language models (LLMs) on the specific tasks assigned to each component. The code for reproducing the experiments is publicly available at: https://github.com/manuel-vg/neurosylogix.

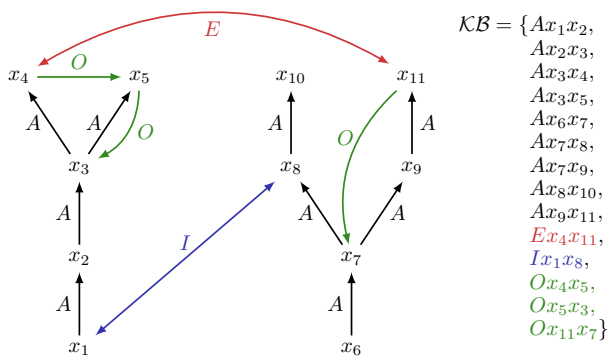
4.1 Synthetic Data

A knowledge base $\mathcal{KB}$ can be formally represented as an edge-labeled graph $G = (V, E, \gamma)$, where the set of vertices V are terms from the domain $\mathcal{X}$ and the set of formulas correspond to the set of edges $E \subseteq \{(u, v) \mid u, v \in V \text{ and } u \neq v\}$ along with a labeling function $\gamma : E \mapsto \mathcal{Q}$ that maps edges to syllogistic quantifiers (see Figure 2 for an example). We produced synthetic knowledge bases by randomly generating graphs such that the resulting knowledge bases are consistent. Furthermore, we imposed the constraint that for every formula F derivable from a given knowledge base $\mathcal{KB}$, there exists a unique subset $\mathcal{P} \subseteq \mathcal{KB}$ such that $\mathcal{P} \vdash F$ is minimal. This property is critical for eliminating redundant derivations of the same hypothesis. We refer to such structures as *non-redundant* knowledge bases.

To convert these structured representations into natural language inputs, terms are replaced with artificially generated pseudowords, and formulas are translated into a textual representation (see Figure 3 for an example).

4.2 Fine-Tuning LLMs

To evaluate the impact of neural assistance on symbolic proof search, we fine-tuned two transformer-based architectures of different sizes: FLAN-T5-base (Raffel et al., 2020), a compact *encoder-decoder* model developed by Google AI, and GPT-4o-mini (OpenAI, 2024), a larger, state-of-the-art *decoder-only* model developed by OpenAI. Importantly, our goal is not to teach general reasoning through fine-tuning, but to adapt models for specific tasks—premise selection and proof by contradiction—while improvements in reasoning skills may still arise. We therefore view our results as reflecting the overall reasoning capabilities of pre-trained and fine-tuned models. In this setting, for both tasks the input



τ	**Examples of syllogisms**
1	$\{Ax_1{-}x_3, Ox_5x_3\} \vdash Ox_5x_1$
2	$\{Ax_6{-}x_{11}\} \vdash Ax_6x_{11}$
3	$\{Ax_2{-}x_4, Ax_7{-}x_{11}, Ax_7x_8, Ex_4x_{11}\} \vdash Ox_8x_2$
4	$\{Ax_7{-}x_{10}, Ax_7{-}x_{11}\} \vdash Ix_{10}x_{11}$
5	$\{Ax_1{-}x_4, Ax_6{-}x_{11}, Ax_8x_{10}, Ex_4x_{11}, Ix_8x_1\} \vdash Ox_{10}x_6$
6	$\{Ax_1{-}x_4, Ax_6{-}x_{11}, Ex_4x_{11}\} \vdash Ex_6x_1$
7	$\{Ax_1{-}x_4, Ax_8x_{10}, Ix_8x_1\} \vdash Ix_4x_{10}$

Figure 2: Example of a knowledge base $\mathcal{KB}$ represented as a graph, along with valid inferences that can be derived from $\mathcal{KB}$.

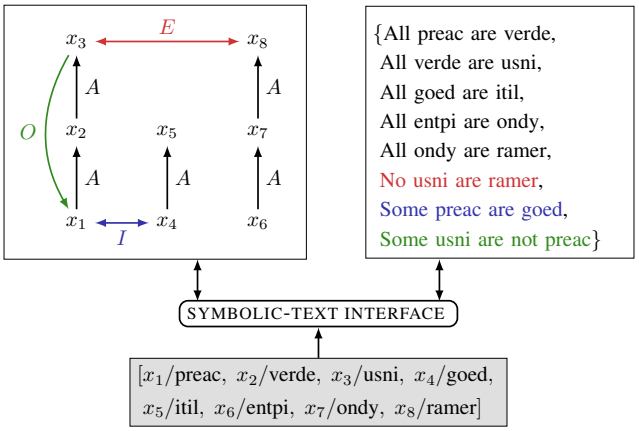


Figure 3: Example of a symbolic–text interface mapping a set of syllogistic formulas, represented as a graph, to a set of natural-language formulas via pseudoword substitution, and vice versa.

consists of a complete knowledge base paired with a hypothesis to be proven, while the output depends on the task: in premise selection, it is the subset of premises required to derive the hypothesis; in proof by contradiction, it is a formula enabling a type (iii) derivation. In both cases, the models process inputs and produce outputs as plain text sequences.

Generalization Criteria We investigate two aspects of generalization in our framework: *compositionality*, the capacity to deconstruct complex structures into simpler components, and *recursiveness*, the ability to iteratively combine simpler structures to construct more complex ones. We operationalize these concepts by systematically excluding syllogisms with short and long *A-chains* during training and evaluating model performance on them exclusively at test time. More precisely (see Figure 4), a model is said to exhibit compositional generalization if it can correctly infer syllogisms with shorter *A-chains* than those encountered during training. Conversely, it demonstrates recursive generalization if it successfully predicts syllogisms with longer *A-chains* than those used in training. To implement this evaluation, we excluded from the training set the five shortest and five longest *A-chain* lengths for each syllogism type.

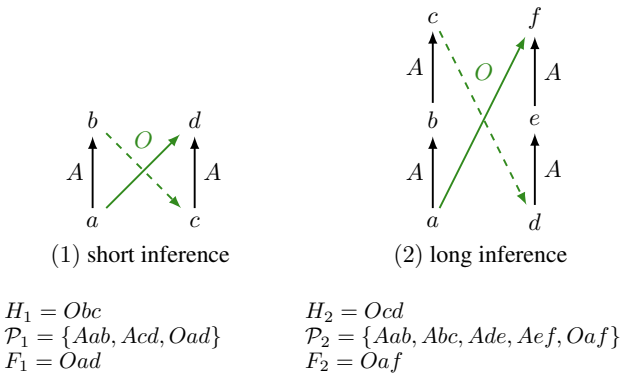


Figure 4: Example of inference-length generalization for a syllogism of type τ_1. Given a knowledge base and a hypothesis H (shown with dashed lines), models predict a subset of premises $\mathcal{P}$ and a refutation formula F for the premise selection and proof by contradiction tasks, respectively. A recursive model trained on (1) generalizes to (2), while a compositional model trained on (2) generalizes to (1).

We first trained models without restrictions on *A-chain* lengths and optimized their accuracy to establish overall performance. These *overall* models serve as the baseline for evaluating generalization, as the *compositional* and *recursive* variants use the same training configuration, differing only in the exclusion of syllogisms with *A-chain* in the designated length ranges.

Data Specification Our dataset comprises 30 distinct synthetic knowledge bases for fine-tuning and an additional 30 for evaluation. To further assess the models' capacity for recursive generalization, we constructed an extra set of 60 KBs, specifically designed to include a greater proportion of inferences involving longer *A-chains*, which are typically underrepresented. On average, each knowledge base contains 40 premises and 1333 valid hypotheses. A breakdown of the mean number of hypotheses per task, categorized by syllogism type, is provided in Table 2. Note that the proof by contradiction task excludes hypotheses that can be derived using only proof types (i) and (ii).

To prevent memorization and enhance generalization, each KB was augmented by applying multiple pseudoword substitutions, and for each substitution, random permutations of the premises were generated. Table 3 summarizes the number of base and augmented KBs, as well as the total

Task	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7	Total
PS	68	152	245	513	42	110	203	1333
PBC	62	–	245	361	42	–	202	911

Table 2: Average number of hypotheses per KB by syllogism type for premise selection (PS) and proof by contradiction (PBC)

	Training	Test (OVE/COM)	Test (REC)
Base KBs	30	30	60
Substitutions	10	3	3
Permutations	3	1	1
Augmented KBs	900	90	180
Total Hypotheses	∼1.1M	∼107K	∼30K

Table 3: Dataset sizes and augmentation settings for the training and evaluation splits; OVE, COM, and REC denote overall, compositional, and recursive models

number of valid hypotheses for each split (training and test). Premise permutations are not included in the test dataset, as models trained with this augmentation generalize perfectly to unseen permutations.

Following KB augmentation, we train all models by subsampling from the total number of hypotheses, stratified by syllogism type and *A-chain* length. T5 required 80% of the data for optimal performance, while GPT achieved comparable results with only 25%, and further increases provided only marginal gains. It is worth noting that 25% still represents a substantial proportion: our GPT models were trained on an average of 100 million tokens[1], even though OpenAI suggests that smaller datasets may suffice for fine-tuning. This highlights the necessity of large-scale data for robust performance in logical reasoning tasks.

5 Empirical Analysis of Generalization in LLMs

As described in the previous section, we construct datasets $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$ for training and evaluating overall models. To study generalization, we derive modified training sets by excluding instances with either the shortest or longest inference lengths from $\mathcal{D}_{\text{train}}$. For evaluation, we construct corresponding test subsets with short and long inference lengths, denoted as $\mathcal{D}^s_{\text{test}}$ and $\mathcal{D}^l_{\text{test}}$, respectively. We design five experimental settings with different training and evaluation splits (see Table 4). Three settings correspond to overall models (OVE), evaluated in-distribution on $\mathcal{D}_{\text{test}}$, $\mathcal{D}^s_{\text{test}}$, and $\mathcal{D}^l_{\text{test}}$, namely OVE-all, OVE-short, and OVE-long, respectively. The remaining two settings correspond to compositional (COM) and recursive (REC) models, evaluated out-of-distribution on $\mathcal{D}^s_{\text{test}}$ and $\mathcal{D}^l_{\text{test}}$, respectively.

Table 5 presents the average accuracy across all evaluated knowledge bases for each experimental setting, using T5 and GPT architectures fine-tuned on premise selection and proof by contradiction tasks. Each experiment was run three times

[1]Fine-tuning cost: ≈ $3.1K (excluding evaluation and preliminary testing)

Experiment	Training	Test
OVE-all	$\mathcal{D}_{\text{train}}$	$\mathcal{D}_{\text{test}}$
OVE-short	$\mathcal{D}_{\text{train}}$	$\mathcal{D}^s_{\text{test}}$
OVE-long	$\mathcal{D}_{\text{train}}$	$\mathcal{D}^l_{\text{test}}$
COM	$\mathcal{D}_{\text{train}} \setminus \mathcal{D}^s_{\text{train}}$	$\mathcal{D}^s_{\text{test}}$
REC	$\mathcal{D}_{\text{train}} \setminus \mathcal{D}^l_{\text{train}}$	$\mathcal{D}^l_{\text{test}}$

Table 4: Training and evaluation splits for each experimental setting

Experiment	Premise Selection		Proof By Contradiction	
	T5	GPT	T5	GPT
OVE-all	0.94 ± 0.05	0.94 ± 0.05	0.93 ± 0.04	0.95 ± 0.04
OVE-short	0.99 ± 0.01	0.98 ± 0.01	0.99 ± 0.01	0.98 ± 0.02
OVE-long	0.79 ± 0.17	0.83 ± 0.15	0.76 ± 0.15	0.88 ± 0.15
COM	0.84 ± 0.04	0.76 ± 0.04	0.67 ± 0.05	0.85 ± 0.05
REC	0.80 ± 0.15	0.82 ± 0.15	0.71 ± 0.18	0.86 ± 0.17

Table 5: Accuracy scores for premise selection and proof by contradiction tasks (all experiments)

for T5 and twice for GPT, and we report the best run to provide a reference for subsequent evaluation of hybrid models.

The OVE family of experiments is designed to characterize generalization to unseen KB structures under seen inference lengths. Results show that OVE-all achieves consistently high accuracy across all tasks and models, while OVE-short and OVE-long reveal a dependence on inference length. We then turn to the COM and REC settings, which test models on both unseen KB structures and unseen inference lengths. In the COM setting, performance drops substantially relative to OVE-short, where accuracy is near perfect. In contrast, in the REC setting, both T5 and GPT perform comparably to OVE-long, but remain below the performance observed in OVE-short.

A more comprehensive analysis is provided in Figure 5, which illustrates the accuracy of evaluated inferences across the five shortest and five longest unseen *A-chain* lengths. For a given syllogism of type τ, we denote the shortest evaluated length as $\sigma(\tau)$ and the longest as $\mu(\tau)$. Solid lines in the plot represent compositional and recursive models, while dashed lines depict overall models. Note that overlapping lines would imply a perfect generalization. The latter is visible in the recursive evaluation for the premise selection task (top-right plot). However, a slight drop in the accuracy occurs as the lengths increase. The drop is more evident for the proof by contradiction task (bottom-right plot). This suggests that LLMs experience difficulties in generating long sequences of A-formulas, regardless of their presence during training. Compositional experiments, on the other hand, exhibit a poor generalization and a steeper curve towards the shortest length, in contrast to overall models that can generate inferences involving shorter *A-chains* almost perfectly.

5.1 Analysis by Syllogism Type

We further analyze these experiments by considering each specific syllogism type (see Table 1) and decomposing the results in Table 5 into a detailed breakdown of accuracy

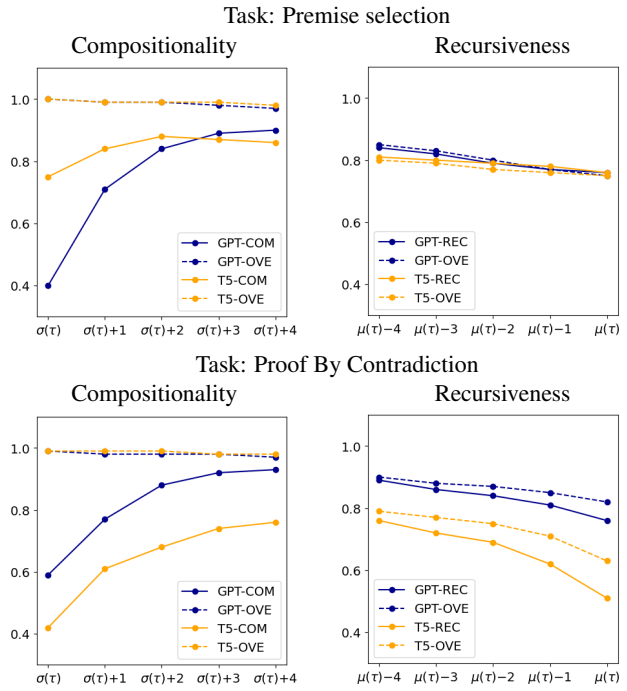


Figure 5: Generalization performance of GPT and T5 architectures across the five shortest and five longest unseen *A-chain* lengths. The y-axis shows accuracy, and the x-axis shows unseen lengths. For each syllogism type τ, $\sigma(\tau)$ and $\mu(\tau)$ denote the shortest and longest lengths considered, respectively.

τ	OVE-all	OVE-short	OVE-long	COM	REC
1	0.90 ± 0.10	0.96 ± 0.05	0.58 ± 0.38	0.35 ± 0.11	0.59 ± 0.35
2	1.00 ± 0.01	1.00 ± 0.00	0.97 ± 0.09	0.99 ± 0.01	0.96 ± 0.08
3	0.85 ± 0.13	0.96 ± 0.05	0.61 ± 0.37	0.88 ± 0.06	0.66 ± 0.33
4	0.97 ± 0.03	1.00 ± 0.01	0.77 ± 0.21	0.87 ± 0.06	0.74 ± 0.23
5	0.72 ± 0.28	0.71 ± 0.36	0.65 ± 0.41	0.49 ± 0.37	0.69 ± 0.38
6	0.97 ± 0.08	0.99 ± 0.03	0.84 ± 0.33	0.90 ± 0.05	0.88 ± 0.28
7	0.96 ± 0.08	1.00 ± 0.01	0.76 ± 0.31	0.77 ± 0.12	0.80 ± 0.28

Table 6: T5 accuracy scores by type (premise selection)

scores for each reasoning task: premise selection and proof by contradiction.

Premise Selection Task The outcomes of the premise selection task are reported in Tables 6 and 7 for the T5 and GPT models, respectively. Notably, type τ_2—the structurally simplest inference embedded within all other types—consistently achieved the highest accuracy across all experimental settings, demonstrating near-perfect generalization, with the unique exception of GPT in the compositionality experiment. In contrast, types τ_1 and τ_5 achieved the lowest accuracies for T5 across the generalization tasks, whereas for GPT, poor performance was observed only on type τ_1.

Proof By Contradiction Task Tables 8 and 9 present the results for proof by contradiction. For this reasoning task—where types τ_2 and τ_6 were not applicable—type τ_7 emerged

τ	OVE-all	OVE-short	OVE-long	COM	REC
1	0.84 ± 0.15	0.91 ± 0.10	0.54 ± 0.36	0.20 ± 0.11	0.52 ± 0.38
2	1.00 ± 0.01	1.00 ± 0.00	0.98 ± 0.06	0.89 ± 0.04	0.97 ± 0.06
3	0.87 ± 0.13	0.98 ± 0.04	0.67 ± 0.36	0.87 ± 0.07	0.68 ± 0.35
4	0.97 ± 0.04	0.99 ± 0.02	0.81 ± 0.22	0.76 ± 0.05	0.78 ± 0.24
5	0.84 ± 0.23	0.93 ± 0.17	0.70 ± 0.38	0.93 ± 0.16	0.68 ± 0.39
6	0.99 ± 0.04	1.00 ± 0.02	0.94 ± 0.19	0.92 ± 0.02	0.92 ± 0.23
7	0.96 ± 0.09	0.99 ± 0.03	0.86 ± 0.25	0.71 ± 0.12	0.86 ± 0.24

Table 7: GPT accuracy scores by type (premise selection)

τ	OVE-all	OVE-short	OVE-long	COM	REC
1	0.97 ± 0.05	0.98 ± 0.04	0.88 ± 0.21	0.23 ± 0.12	0.82 ± 0.27
3	0.88 ± 0.09	0.98 ± 0.04	0.54 ± 0.33	0.95 ± 0.05	0.37 ± 0.33
4	0.97 ± 0.03	1.00 ± 0.00	0.78 ± 0.20	0.66 ± 0.06	0.77 ± 0.22
5	0.78 ± 0.29	0.68 ± 0.39	0.78 ± 0.33	0.36 ± 0.36	0.84 ± 0.31
7	0.95 ± 0.07	0.99 ± 0.03	0.85 ± 0.25	0.83 ± 0.12	0.80 ± 0.29

Table 8: T5 accuracy scores by type (proof by contradiction)

as the best-performing inference type on both architectures, achieving a near-perfect generalization in the case of GPT. Overall, this task appeared to be easier for GPT than for T5, which, with few exceptions, struggled to generalize effectively. GPT, by contrast, demonstrated consistently strong performance across all inference types, including the particularly challenging type τ_1.

5.2 Analysis of Incorrect Predictions

We first analyze the types of errors that arise in the premise selection task. In particular, we distinguish between cases where the model includes unnecessary premises, still allowing the construction of valid proofs, and cases where there are missing premises, representing true prediction failures.

Unnecessary Premises We have found that models occasionally predict unnecessary premises—i.e., premises that are not strictly required to derive the target hypothesis. While such predictions are technically incorrect with respect to minimal inferences, they nonetheless allow to construct a valid proof of the hypothesis. Therefore, they remain acceptable inputs for the symbolic prover.

We performed an analysis by treating such predictions as correct and report the resulting accuracy gain, defined as the difference between non-minimal and minimal accuracies in the generalization experiments. Tables 10 and 11 present the results for the compositional and recursive settings, respectively. We additionally report the mean and standard deviation of the number of unnecessary premises predicted by each model. In the compositional setting, T5 shows only

τ	OVE-all	OVE-short	OVE-long	COM	REC
1	0.97 ± 0.06	0.96 ± 0.07	0.93 ± 0.20	0.90 ± 0.14	0.90 ± 0.24
3	0.92 ± 0.12	0.97 ± 0.08	0.81 ± 0.30	0.97 ± 0.05	0.74 ± 0.34
4	0.97 ± 0.04	0.99 ± 0.02	0.87 ± 0.19	0.78 ± 0.05	0.81 ± 0.23
5	0.85 ± 0.30	0.78 ± 0.37	0.96 ± 0.16	0.73 ± 0.37	0.93 ± 0.22
7	0.97 ± 0.07	0.98 ± 0.06	0.92 ± 0.19	0.98 ± 0.04	0.95 ± 0.15

Table 9: GPT accuracy scores by type (proof by contradiction)

Length	T5		GPT	
	Gain	# Unnec. Prem.	Gain	# Unnec. Prem.
$\sigma(\tau)$	0.08	5.55 ± 4.66	0.26	5.51 ± 3.53
$\sigma(\tau)+1$	0.02	5.77 ± 4.11	0.10	5.33 ± 3.86
$\sigma(\tau)+2$	0.00	4.76 ± 4.10	0.04	5.28 ± 3.63
$\sigma(\tau)+3$	0.01	4.40 ± 3.79	0.02	5.31 ± 3.78
$\sigma(\tau)+4$	0.00	5.22 ± 4.87	0.02	5.30 ± 3.94
Total	0.02	5.44 ± 4.48	0.09	5.42 ± 3.66

Table 10: Accuracy gain from non-minimal proofs and number of unnecessary premises for shorter unseen lengths (COM)

Length	T5		GPT	
	Gain	# Unnec. Prem.	Gain	# Unnec. Prem.
$\mu(\tau)-4$	0.00	5.21 ± 3.48	0.02	4.22 ± 4.22
$\mu(\tau)-3$	0.01	6.86 ± 3.58	0.02	4.21 ± 4.16
$\mu(\tau)-2$	0.00	7.46 ± 3.77	0.02	3.48 ± 3.47
$\mu(\tau)-1$	0.00	2.75 ± 1.30	0.02	3.46 ± 3.69
$\mu(\tau)$	0.00	5.67 ± 1.25	0.01	3.86 ± 4.69
Total	0.00	6.06 ± 3.63	0.02	3.98 ± 4.04

Table 11: Accuracy gain from non-minimal proofs and number of unnecessary premises for longer unseen lengths (REC)

a marginal improvement of $\approx 2\%$, whereas GPT exhibits a substantial gain of $\approx 9\%$. In the recursive setting, T5 shows no improvement, while GPT achieves a modest gain of $\approx 2\%$. Regarding unnecessary premises, in the compositional setting both T5 and GPT predict a similar average number (≈ 5), whereas in the recursive setting T5 predicts slightly more (≈ 6) than GPT (≈ 4).

Missing Premises Finally, we examined incorrect model predictions that did not involve the inclusion of unnecessary premises, in order to better understand the inferential patterns being captured. In all such cases, the models consistently generated well-formed formulas. Therefore, the observed errors can be attributed to the incorrect selection or construction of syllogistic formulas, rather than to syntactic malformation.

To evaluate semantic errors in the premise selection task, we considered three key aspects: *term overlap*, assessing whether the terms appearing in the hypothesis were also present in the predicted premises, which would indicate a basic understanding of syllogistic rules; *premise validity*, examining whether the predicted premises were sourced from the knowledge base; and *term validity*, verifying whether the terms within the predicted premises were restricted to those found in the knowledge base. These latter checks ensure that the models are not generating fabricated content.

Table 12 presents a summary of semantic validity by reporting the proportion of cases, across all experiments, in which each criterion was satisfied for both T5 and GPT models. These proportions were calculated relative to the set of incorrect predictions. Interestingly, the results indicate that the models generate fabricated premises. A closer analysis suggests that this issue may stem from confusion between syllogisms whose hypotheses share the same formula

Model	Exp.	Term overlap	Prem. valid.	Term val.
T5	OVE	0.77	0.35	0.92
	COM	0.46	0.25	0.90
	REC	0.63	0.18	0.89
GPT	OVE	0.94	0.20	0.92
	COM	0.71	0.31	0.94
	REC	0.92	0.32	0.92

Table 12: Semantic validity for the premise selection task

type—specifically, among O-formulas in inference types τ_1, τ_3, and τ_5, and among I-formulas in types τ_4 and τ_7. That is, the models appear to erroneously construct an incorrect type of syllogism, thereby generating fabricated premises in an attempt to force a valid inference. We illustrate this phenomenon with two representative cases:

- First, consider syllogisms of type τ_1. To derive Obc, the required premises are $\{Aa-b, Ac-d, Oad\}$. However, the model attempts to construct a proof of type τ_3 instead, which requires the premises $\{Aa–b, Ac–d, Aa–e, Ede\}$. In doing so, the model predicts Ede instead of the required Oad and assumes the existence of $Aa-e$.
- Second, consider syllogisms of type τ_4, where the goal is to derive Ibc from the premises $\{Aa–b, Aa–c\}$. Here, the model sometimes attempts to apply type τ_7, which has the form $\{Aa-b, Ac-d, Iac\} \vdash Ibd$. In this case, the model predicts a formula of the form Iax and assumes a formula $Ax-c$ that does not exist.

As a result, in both cases, the model may fabricate premises in order to complete the incorrectly assumed proof structure.

Incorrect Formulas in the Proof by Contradiction Task Similarly, in the proof by contradiction task, all incorrect predictions were syntactically well-formed. While the criteria concerning term overlap with the hypothesis and the validity of premises are not directly applicable in this setting, it is noteworthy that, in all instances, the terms used in the incorrectly predicted formulas were contained within the vocabulary of the knowledge base (*term validity*).

6 Neuro-Symbolic Reasoning Framework

To formalize the integration of symbolic and connectionist components, we represent the neural networks as functions that map a hypothesis and a knowledge base to a set of syllogistic formulas:

$$\mathrm{M_{PS}} : (H, \mathcal{KB}) \mapsto 2^{\mathcal{F}} \qquad \mathrm{M_{PBC}} : (H, \mathcal{KB}) \mapsto 2^{\mathcal{F}}.$$

Here, $\mathrm{M_{PS}}$ denotes the *premise selection* model, while $\mathrm{M_{PBC}}$ denotes the *proof by contradiction* model. Both neural components are designed to provide assistance to a symbolic prover. An overview of the resulting hybrid framework is illustrated in Figure 1. In the remainder of this section, we present a simple algorithm that implements this integration, followed by an experimental evaluation assessing the behavior and performance of the proposed hybrid models.

Algorithm 4 Hybrid Syllogistic Prover

Input: A hypothesis H and a knowledge base $\mathcal{KB}$.
Output: A proof $\triangledown$ of H from $\mathcal{KB}$ (if H is valid).

1: $\mathcal{P} \leftarrow \mathrm{M_{PS}}(H, \mathcal{KB})$
2: $\mathcal{S} \leftarrow \mathrm{M_{PBC}}(H, \mathcal{KB})$
3: $\mathcal{C} \leftarrow \langle \textsc{cp}(\mathcal{S}) \rangle \| \langle \textsc{all\_pairs}(\mathcal{Q}, \mathcal{X}) \setminus \textsc{cp}(\mathcal{S}) \rangle$
4: $\triangledown \leftarrow$ SYLLOGISTIC PROVER$(H, \mathcal{P})$
5: **if** $\triangledown \neq$ `False` **then**
6: **return** $\triangledown$
7: **end if**
8: **return** SYLLOGISTIC PROVER$(H, \mathcal{KB})$

6.1 Integration of Symbolic and Connectionist Components

The hybrid model reuses the symbolic algorithms from Section 3 while incorporating predictions from neural assistants to restrict the search space and reduce non-deterministic exploration during inference. Algorithm 4 summarizes the resulting hybrid syllogistic prover. Given a knowledge base $\mathcal{KB}$ and a hypothesis H, the neural components $\mathrm{M_{PS}}$ and $\mathrm{M_{PBC}}$ first predict the necessary premises to prove H, and formulas to prove H by contradiction, respectively. These predictions are used to initialize the symbolic search (lines 1-2). In particular, the set of contradictory pairs $\mathcal{C}$ (cf. Algorithm 3, line 1) is initialized with $\textsc{cp}(\mathcal{S}) = \{(F, \overline{F}) \mid F \in \mathcal{S}\}$ so that candidate pairs suggested by the neural assistant are explored first (line 3). The symbolic prover (Algorithm 1) is then executed on a restricted subset $\mathcal{P} \subseteq \mathcal{KB}$ (line 4). If no proof is found, the prover is called again using the full knowledge base (line 8), thereby reverting to the purely symbolic setting.

Two properties of the framework are worth noting. First, if the *premise selection* model $\mathrm{M_{PS}}$ fails to provide the premises necessary for deriving the hypothesis, the procedure restarts without restrictions, ensuring completeness at the cost of additional computation. Second, the *proof by contradiction* model $\mathrm{M_{PBC}}$ may affect only the order in which contradictory pairs are explored; even if its predictions are ineffective, the non-deterministic nature of the search guarantees no impact on correctness and does not require additional computation. Thus, neural assistance influences only efficiency, not correctness, and the hybrid prover is guaranteed to terminate and to remain sound and complete regardless of neural performance.

6.2 Experimental Evaluation

We investigated three variants of hybrid models, each incorporating neural components trained under different generalization regimes—namely, overall, compositional, and recursive—and evaluated their performance relative to a purely symbolic baseline to assess the impact of the neural assistants.

Sampling Procedure We randomly selected just over 2000 samples from 30 distinct knowledge bases within the test dataset. From each knowledge base, we selected 10 syllogisms of each type, comprising 5 instances featuring longer *A-chains* and 5 with shorter *A-chains*. To ensure a sufficient level of inference complexity, we excluded trivial proofs by retaining only those syllogisms in which the *A-chains* length is greater than or equal to 2.

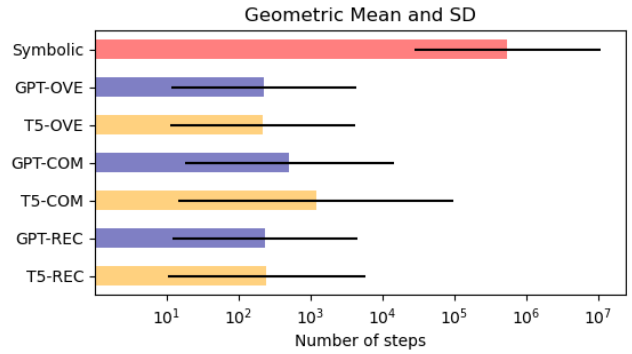


Figure 6: Geometric mean and standard deviation of the number of steps for the symbolic and hybrid models using different assistants trained on GPT and T5. OVE, COM, and REC denote overall, compositional, and recursive models, respectively.

Symbolic vs. Hybrid Evaluation To compare the efficiency of symbolic and hybrid models, we measured the number of steps required to complete the proof of a valid hypothesis, where each step corresponds to a single invocation of the recursive function DRV by the prover. Due to the non-deterministic nature of the symbolic component, each experiment is executed five times for each model configuration. Moreover, we use logarithmic notation and the geometric mean—due to the potentially wide variability— to represent step counts (see Figure 6).

On average, the symbolic model requires approximately $10^{5.7}$ steps to complete a proof. In contrast, hybrid models require substantially fewer steps. In particular, models incorporating overall and recursive neural assistants, exhibit comparable performance, and they need only around $10^{2.4}$ steps across both LLM architectures. This corresponds to a reduction of approximately three orders of magnitude. Hybrid configurations assisted by compositional models require slightly more steps, approximately $10^{2.7}$ for the GPT-based model and $10^{3.1}$ for the T5-based model. This outcome is not unexpected, as they exhibit a drop in accuracy relative to their overall and recursive counterparts. However, their use in hybrid configurations does not result in a significant increase in derivation steps, indicating robustness when assisting the prover.

7 Conclusions

Our main findings focus on the performance of Large Language Models (LLMs) and their role in assisting automated provers. From a semantic perspective, LLMs struggle to fully grasp logical reasoning. Our generalization experiments highlight a significant gap between recursiveness and compositionality, indicating a need for a deeper theoretical

understanding of this issue. One possible explanation is that recursiveness relies on repetitive pattern matching, whereas compositionality requires structural decomposition—a process that transformer attention mechanisms may not naturally support. To better understand this gap, we provide a more fine-grained analysis across different syllogism types where we observed notable differences in performance and generalization. In particular, language models can reason about single *A-chains* (syllogisms of type τ_2), whereas types combining multiple *A-chains* with additional syllogistic formulas pose greater challenges. These findings open a research agenda for understanding how different reasoning building blocks affect model performance, including in more complex fragments of logic, where new generalization patterns may appear.

We performed a systematic analysis to investigate the models' incorrect outputs. Two patterns of interest were noted: models occasionally predict unnecessary premises while still including the correct ones, and sometimes confuse syllogisms whose hypotheses share the same formula type, producing fabricated premises. These observations motivate further investigation into the sources of such errors and the development of methods to mitigate them.

We consider a relatively small encoder–decoder model (T5), chosen for its efficiency and strong overall performance on our tasks, alongside a substantially larger decoder-only model (GPT), selected to evaluate a state-of-the-art LLM. GPT shows greater efficiency, converging with less data, though both models achieve comparable performance. This suggests that scaling to larger models alone may not be sufficient to overcome the challenges posed by these reasoning tasks. Nevertheless, our results demonstrate that neither the limitations in generalization nor model size prevent LLMs from effectively assisting symbolic provers. On the contrary, this assistance fosters a collaborative relationship between connectionist and symbolic models. While connectionist models can simplify and expedite tasks, symbolic models can still complete them when necessary, making hybrid models an important area for further investigation.

This study focuses on syllogistic logic, one of the simplest fragments of formal reasoning. Despite its limited expressive power, it already exposes non-trivial challenges for generalization. Future work will aim to explain why certain reasoning types are learnable while others are not, and to clarify whether these limitations are grounded in structural aspects of logical composition. To further investigate this question, we will extend this analysis to richer fragments, e.g., Pratt-Hartmann (2004) or suitable fragments of modal logic. Moreover, our modular approach could provide practical solutions for developing computationally efficient provers for these logics, forming part of a broader effort to determine where the boundary of tractability lies for neural and neuro-symbolic reasoners.

Acknowledgments

This research was funded by the National Science Center, Poland under the OPUS call [grant 2020/37/B/HS1/04220]. We gratefully acknowledge Polish high-performance computing infrastructure PLGrid (HPC Center: ACK Cyfronet AGH) for providing computer facilities and support within computational grant no. PLG/2024/017165.

AI Declaration

The authors have not employed any Generative AI tools.

References

Bertolazzi, L.; Vargas Guzmán, M.; Bernardi, R.; Malicki, M.; and Szymanik, J. 2026. Teaching small language models to learn logic through meta-learning. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, 8049–8080.

Bertolazzi, L.; Gatt, A.; and Bernardi, R. 2024. A systematic analysis of large language models as soft reasoners: The case of syllogistic inferences. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 13882–13905.

Huang, J., and Chang, K. C.-C. 2023. Towards reasoning in large language models: A survey. In *Findings of the Association for Computational Linguistics: ACL 2023*, 1049–1065.

Hupkes, D.; Giulianelli, M.; Dankers, V.; Artetxe, M.; Elazar, Y.; Pimentel, T.; Christodoulopoulos, C.; Lasri, K.; Saphra, N.; Sinclair, A.; et al. 2023. A taxonomy and review of generalization research in nlp. *Nature Machine Intelligence* 5(10):1161–1174.

Janssen, T. M. V., and Partee, B. H. 1997. Compositionality. In *Handbook of Logic and Language*. Elsevier. 417–473.

Lake, B. M., and Baroni, M. 2023. Human-like systematic generalization through a meta-learning neural network. *Nature* 623(7985):115–121.

Lake, B. M.; Ullman, T. D.; Tenenbaum, J. B.; and Gershman, S. J. 2017. Building machines that learn and think like people. *Behavioral and Brain Sciences* 40:e253.

Marcus, G. 2018. Deep learning: A critical appraisal. *arXiv preprint arXiv:1801.00631*.

Mondorf, P., and Plank, B. 2024. Beyond accuracy: Evaluating the reasoning behavior of large language models - a survey. *arXiv preprint arXiv:2404.01869*.

OpenAI. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Pratt-Hartmann, I. 2004. Fragments of language. *Journal of Logic, Language and Information* 13(2):207–223.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research* 21(140):1–67.

Smiley, T. J. 1973. What is a syllogism? *Journal of Philosophical Logic* 2(1):136–154.

Vargas Guzmán, M.; Szymanik, J.; and Malicki, M. 2024. Testing the limits of logical reasoning in neural and hybrid models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, 2267–2279.