When No Paths Lead to Rome: Benchmarking Systematic Neural Relational Reasoning

Anirban Das[*]
Cardiff University
dasa8@cardiff.ac.uk

Irtaza Khalid[*]
Cardiff University
khalidmi@cardiff.ac.uk

Rafael Peñaloza
University of Milano-Bicocca
rafael.penalozanyssen@unimib.it

Steven Schockaert
Cardiff University
schockaerts1@cardiff.ac.uk

Abstract

Designing models that can learn to reason in a systematic way is an important and long-standing challenge. In recent years, a wide range of solutions have been proposed for the specific case of systematic relational reasoning, including Neuro-Symbolic approaches, variants of the Transformer architecture, and specialised Graph Neural Networks. However, existing benchmarks for systematic relational reasoning focus on an overly simplified setting, based on the assumption that reasoning can be reduced to composing relational paths. In fact, this assumption is hard-baked into the architecture of several recent models, leading to approaches that can perform well on existing benchmarks but are difficult to generalise to other settings. To support further progress in the field of systematic relational reasoning with neural networks, we introduce NoRA, a new benchmark which adds several levels of difficulty and requires models to go beyond path-based reasoning.

1 Introduction

The problem of *relational reasoning* involves predicting relationships between entities that are entailed from a given set of facts (expressing properties of different entities and how they are related). Entailment arises from a set of rules that a model must learn from examples. The central challenge lies in designing models capable of *systematic* reasoning, a concept closely linked to compositional generalization [Hupkes et al., 2020]. This means that models should be able to solve test cases by applying the rules they have learned in novel ways. Recently, various neural network models have been proposed for this purpose, including neuro-symbolic approaches [Minervini et al., 2020], path-based methods [Cheng et al., 2023b], transformer variants [Bergen et al., 2021], and graph neural networks (GNNs) [Khalid and Schockaert, 2025].

Two significant problems are the lack of datasets that adequately test for systematicity, and the fact that state-of-the-art models heavily leverage the structure of existing benchmarks. CLUTRR [Sinha et al., 2019], the most popular benchmark for assessing systematicity, focuses on inferring family relationships. While all CLUTRR training examples can be solved in at most four inference steps, the test examples may require up to ten. Standard GNNs struggle with this kind of length generalization. Furthermore, the most successful neural methods exploit a specific characteristic of CLUTRR: the reasoning process reduces to composing relations along a single path connecting the target and source

---

[*]Equal contribution.

```
wes is the grandparent of todd
wes is daughter-less
⇒ wes is the paternal grandparent of todd

ann is an aunt of todd
wes is daughter-less
⇒ ann is not from wes's side of the family
⇒ ann is the maternal aunt of todd
```
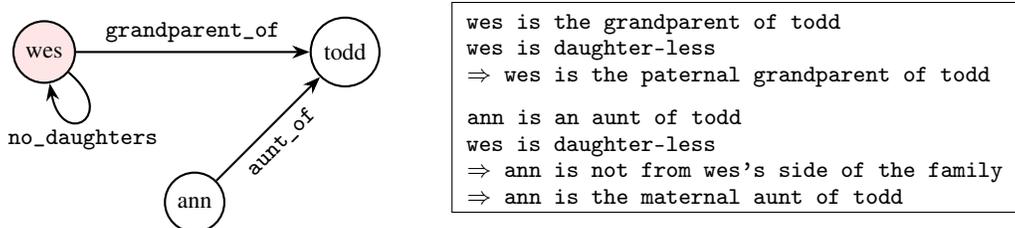
Figure 1: Example where path-based reasoning fails: to derive that `ann` is `todd`'s maternal aunt, one must consider `wes`, who is not on any connecting path between `ann` and `todd`.

entities, where the relational facts are viewed as a knowledge graph. For example, given the path $a \xrightarrow{\textit{brother-of}} b \xrightarrow{\textit{daughter-of}} c \xrightarrow{\textit{brother-of}} d$, one can infer that d is the uncle of a by composing the relations *brother-of*, *daughter-of*, and *brother-of*. We refer to this style of reasoning as *path-based reasoning*. Relational reasoning often requires going beyond path-based reasoning, but this is not reflected in existing benchmarks. The only exception is STaR [Khalid and Schockaert, 2025], which focuses on temporal and spatial reasoning, and requires combining the predictions of multiple relational paths. However, the main style of reasoning that is tested by this benchmark is still path-based.

In this paper, we introduce **NoRA** (Non-Path Reasoning with Ambiguous Facts), a new benchmark which challenges state-of-the-art neural models for relational reasoning. NoRA is inspired by CLUTRR, but it intentionally breaks many of the structural assumptions in CLUTRR that state-of-the-art models are hard-coded to exploit. Like CLUTRR, the rules to be learned in NoRA are intuitive and grounded in everyday relationships—ones that humans and large language models (LLMs) naturally accept as plausible or true. However, NoRA differs from CLUTRR in three key ways.

First, NoRA is specifically designed to break the path-based inductive bias that many existing relational reasoning models rely on. To this end, NoRA considers a richer set of relationships, including more fine-grained, gender-specific family roles such as *maternal aunt of*, and everyday relations such as *is schoolmates with* and *lives in the same place as*, which often require models to go beyond path-based reasoning. Figure 1 illustrates such a case. In this example, we can infer that `ann` is the maternal aunt of `todd`, as explained in the figure, but to arrive at this conclusion, the reasoning must detour through the node `wes`, which is not on a path between `ann` and `todd` in the graph.

A second notable feature of NoRA is that multiple relationships may hold between a given pair of entities. These may be hierarchical (e.g. `ann` is both the `aunt` and the `maternal_aunt` of `todd`) or independent (e.g. a person's `brother` can also be their `schoolmate`).

Finally, NoRA incorporates a small number of ambiguous facts in its problem instances, for instance expressing that a is the `father_of` either b or c. We argue that neural relational reasoning models should be equipped to handle such ambiguity, given its ubiquity in real-world text-based reasoning. To resolve ambiguities, a model must learn to reason with constraints: the model must evaluate multiple possibilities and then (i) eliminate any possibilities that violate constraints and (ii) determine whether a given relationship holds across all the remaining possibilities.

We make the following contributions:

- We introduce NoRA, a benchmark for systematic neural relational reasoning.
- We measure the difficulty of NoRA problem instances along a number of dimensions, corresponding to the length of inference chains, the amount of ambiguity, and the extent to which the required form of reasoning goes beyond path-based reasoning.
- We empirically show that state-of-the-art neural models for systematic reasoning struggle on NoRA, highlighting the need for new approaches.

2 Related work

The problem of learning to reason has traditionally been studied in Inductive Logic Programming [Muggleton and Raedt, 1994] (ILP). Formally, given a background theory $\mathcal{B}$ and sets of positive and negative examples, ILP considers the problem of finding a set of clauses $\mathcal{H}$ such that $\mathcal{B} \cup \mathcal{H}$ logically

entails every positive example and none of the negative examples. While important contributions in ILP continue to be made [Cropper et al., 2022], in recent years the focus has mostly shifted to neuro-symbolic methods, which try to solve the problem of learning to reasoning with a differentiable objective, for instance by simulating logic programming using tensor multiplication [Yang et al., 2017, Sadeghian et al., 2019, Dong et al., 2019], by interpreting logical connectives using fuzzy logic [Evans and Grefenstette, 2018, Sourek et al., 2018, Badreddine et al., 2022], or by using a probabilistic semantics [Manhaeve et al., 2021]. However, these approaches are mostly designed for injecting background knowledge into the training process of a neural network model, or for making one-off predictions (e.g. for knowledge graph completion), rather than for systematic reasoning.

Systematic reasoning tasks require models to learn to compose logical rules to infer conclusions. The difficulty stems from the fact that the derivations (i.e. the specific sequences of rule applications) that are needed for solving test instances differ from those in the training data, even if the training data contains sufficient information to learn all the required rules individually. Existing benchmarks that test for systematic reasoning include CLUTRR [Sinha et al., 2019], which involves predicting family relationships, GraphLog [Cohen, 2019], which involves reasoning about synthetically generated knowledge graphs, and STaR [Khalid and Schockaert, 2025], which involves qualitative temporal and spatial reasoning. CLUTRR and GraphLog can be solved by path-based reasoning, i.e. the target relationship can be inferred by selecting a single relational path between the two target nodes and composing the relations along that path. STaR requires models to compose relationships among multiple paths and then taking the intersection of the resulting predictions. Reasoning is thus more involved, although still mainly path-based. Nonetheless, these benchmarks are already challenging for most approaches. Conditional Theorem Provers (CTPs) [Minervini et al., 2020] were one of the first methods to achieve a near-perfect accuracy on CLUTRR. CTPs use a form of differentiable logic programming based on a soft unification mechanism. An important drawback of CTPs is that they are computationally expensive, which makes them impractical for many applications. Recently, a number of more efficient approaches for systematic reasoning have been proposed, such as R5 [Lu et al., 2022] and NCRL [Cheng et al., 2023a]. For benchmarks that only require path-based reasoning, these approaches can be effective, but they cannot be used in more general settings such as STaR and our proposed benchmark. Edge transformers [Bergen et al., 2021] are a modification of the transformer architecture, with a triangular attention mechanism that is designed to facilitate relational reasoning. They perform well on path-based benchmarks such as CLUTRR, albeit somewhat worse than CTPs, R5 and NCRL. In contrast to the aforementioned methods, their architecture does not constrain them to path-based reasoning. They also performed reasonably well on STaR. Finally, EpiGNNs [Khalid and Schockaert, 2025] are a type of GNN model with an inductive bias for systematic relational reasoning. Their architecture is designed to support reasoning tasks where the predictions of multiple paths need to be combined, and are thus well-suited to benchmarks such as STaR.

The problem of systematic relational learning is fundamentally different from knowledge graph (KG) completion, despite the close similarities in the format of both tasks. KG completion often requires making predictions that cannot be logically entailed, by exploiting statistical biases. Because KG completion models have to capture such biases, they typically perform poorly on systematic generalization tasks. Conversely, models that are designed for systematic reasoning tend to underperform on KG completion benchmarks; see e.g. the comparison between NBFNet [Zhu et al., 2021] and EpiGNN by Khalid and Schockaert [2025]). Interestingly, the fact that path-based reasoning is not always sufficient has also been highlighted in the context of KG completion [Sun et al., 2024].

3 Problem setting

Before introducing the NoRA benchmark, we introduce the problem setting and some notations.

Stories We consider the problem of reasoning about *stories*, which in this context are sets of facts. Stories may contain three types of facts. First, we have binary facts, expressing a relationship between two entities, e.g. school_mates_with(ram, irfan). Second, we have unary facts, e.g. underage(ryan), expressing a property of a single entity. Finally, we also have facts encoding ambiguous relationships. We use the syntax of Answer Set Programming (ASP [Gelfond and Lifschitz,

1988]) to encode such facts.[2] The general form of an ambiguous fact is as follows:

$$l \{r_1(x_1, y_1), ..., r_n(x_n, y_n)\} u$$

It expresses that between l and u of the binary facts $r_1(x_1, y_1), ..., r_n(x_n, y_n)$ are true. We will specifically use such facts to encode relationships where one of the arguments is ambiguous, e.g.:

$$1 \{r(x, y_1), r(x, y_2)\} 1 \tag{1}$$

This encodes that x is in relationship r with either y_1 or y_2 (not both). Such ambiguities often arise when reasoning about information coming from text, for instance because of ambiguous coreferences.

World rules All the stories in our dataset satisfy some regularities, which are formalized using definite rules and constraint rules. We will together refer to them as the *world rules* and again use ASP syntax. *Definite rules* allow us to infer relational facts from a given set of facts, e.g.:

living_in_same_place(X,Z) :- *living_in_same_place(X,Y), living_in_same_place(Y,Z).*

underage(X) :- *school_mates_with(X,U).*

Uppercase arguments like X denote variables. The head (left side of a rule) specifies what is inferred, while the body (right side) specifies the conditions. The first rule expresses that the `living_in_same_place` relation is transitive. The second rule expresses that if somebody is school mates with somebody else, then they must be underage. *Constraint rules* specify that some sets of facts can never be true at the same time. They are encoded as rules with an empty head, e.g.:

:- *underage(X), parent_of(X, Y).*

This constraint expresses that underage people cannot be parents.

Answer sets The world rules allow us to reason about the facts that are specified in a given story $\mathcal{S}$. This process serves two purposes. First, the facts in the story are incomplete, in the sense that we can infer additional facts by applying the definite rules. Second, the constraints allow us to eliminate some of the ambiguity. To formally define the reasoning process, we need the concept of *answer set*.[3] For a story $\mathcal{S}$ without ambiguity, its answer set contains all facts inferrable from $\mathcal{S}$ via definite rules. If constraints are violated by this set, $\mathcal{S}$ has no answer set; otherwise, $\mathcal{A} = ans(\mathcal{S})$ denotes $\mathcal{S}$'s answer set. Now consider a story $\mathcal{S}$ that contains the ambiguous fact (1). There are two possibilities: either $r(x, y_1)$ is true or $r(x, y_2)$ is true. Accordingly, we may consider two alternatives: the story $\mathcal{S}'$ in which the ambiguous fact (1) is replaced by $r(x, y_1)$, and the story $\mathcal{S}''$ in which the ambiguous fact is instead replaced by $r(x, y_2)$. We can repeat this process for all the ambiguous facts, leading to a set of unambiguous stories $\mathcal{S}_1, ..., \mathcal{S}_k$. We will refer to these stories as the *refinements* of $\mathcal{S}$. Then we say that $\mathcal{A}$ is an answer set of $\mathcal{S}$ if there is an $i \in \{1, ..., k\}$ such that $\mathcal{A} = ans(\mathcal{S}_i)$. A story with ambiguous facts may thus have 0, 1 or multiple answer sets. Let us write $ref^+(\mathcal{S})$ for the refinements of $\mathcal{S}$ which have an answer set (i.e. the different ways in which the ambiguities can be resolved without violating any constraints) and let $ref^-(\mathcal{S})$ denote the other refinements (i.e. those where the inferred facts violate some constraints).

Problem formulation The training data consists of tuples $(\mathcal{S}, x, y, \mathcal{R})$, where $\mathcal{S}$ is a story, x and y are entities that appear in $\mathcal{S}$, and $\mathcal{R}$ is the set of relationships that can be inferred to hold between x and y. Formally, let us write $rels(x, y, \mathcal{A})$ for the set of relationships that are asserted to hold between x and y in a given answer set $\mathcal{A}$:

$$rels(x, y, \mathcal{A}) = \{r \mid r(x, y) \in \mathcal{A})$$

then we have:

$$\mathcal{R} = \bigcap \{rels(x, y, \mathcal{A}) \mid \mathcal{A} \text{ is an answer set of } \mathcal{S}\}$$

The dataset is generated such that every story $\mathcal{S}$ has at least one answer set, i.e. there is always a way to resolve the ambiguities which is consistent with the constraints of the world. Test instances are queries of the form $(\mathcal{S}, x, y, ?)$, i.e. given a story and two designated entities x (source) and y (target), the task is to predict all relationships that can be inferred to hold between x and y. The world rules are fixed across all training and test examples. The model is thus required to induce the world rules from the training examples, and to learn to apply them in a systematic way.

---

[2]More precisely, we use the syntax of Clingo: `https://github.com/potassco`.

[3]In general, answer sets are defined in terms of the so-called Gelfond-Lifschitz reduct [Gelfond and Lifschitz, 1988]. For the simplified setting here, answer sets can be defined more straightforwardly.

Example   Suppose we have a story $\mathcal{S}$ consisting of the following facts. [4]:

child_of(john,mary)   *colleague_of(mary,bob)* $1\{$*living_in(bob,paris)*, *living_in(bob,rome)*$\}1$

living_in(john,rome)   *school_mate_with(john,eve)* $1\{$*child_of(eve,ann)*, *child_of(eve,paul)*$\}1$

There are two ambiguous facts, which means that there may be up to four answer sets. However, from *school_mate_with(john,eve)* we infer that *john* is underage. We have a world rule which states that underage children live in the same place as their parents. Together with *living_in(john,rome)* we infer *living_in(mary,rome)*. We have a rule that colleagues live in the same place, allowing us to infer *living_in(bob,rome)*. The option *living_in(bob,paris)* is thus not consistent with the available facts (we have a constraint stating that people cannot live in two different places). The other ambiguity cannot be resolved, so the story has two answer sets. For the query $(\mathcal{S}, mary, rome, ?)$ the answer is thus $\mathcal{R} = \{$*living_in*$\}$, as *living_in(mary,rome)* is included in both answer sets. For the query $(\mathcal{S}, eve, ann, ?)$ the answer is $\mathcal{R} = \emptyset$, as *child_of(eve,ann)* is only included in one of the answer sets.

4   Dataset construction

We now present the details of our benchmark and introduce a number of metrics for measuring different aspects of problem difficulty. These difficulty measures are then used for creating systematic test splits, which will allow us to evaluate different aspects of compositional generalization.

4.1   Data generation process

We generate a story by randomly generating story facts. We use Clingo 5.7.1 [Gebser et al., 2011] to obtain the answer sets. An *entailed atom* is an atom that appears in all the answer sets of the story but is not explicitly provided as a story fact. The entailed atoms are used to construct the queries in our benchmark. We only retain stories that have at least one answer set and at least one entailed atom. We generate many stories, where each story includes a different set of story facts, while the world rules remain constant across all generated stories for a dataset. Details regarding the exact world rules, types of ambiguous facts considered, and the sampling process are in Appendix D.

4.2   Measuring problem difficulty

We propose a number of metrics for measuring the difficulty of a given problem instance. These metrics serve two purposes. First, since we want to test for systematicity, we will consider test instances that are strictly harder than the training instances. To solve such test instances, models need to learn to compose the knowledge they have learned in novel ways (rather than learning shortcuts or memorizing computation graphs). Second, the proposed difficulty metrics will allow us to analyze model performance in a more fine-grained way.

Reasoning depth   A standard notion of difficulty is the number of inference steps that are needed to infer the answer (i.e. the number of rule applications). Let $\mathcal{S}$ be a given story, and let $\mathcal{S}_1, ..., \mathcal{S}_k$ be the refinements of $\mathcal{S}$ that are consistent with the constraints. The answer sets of $\mathcal{S}$ are then given by $\mathcal{A}_1, ..., \mathcal{A}_k$ with $\mathcal{A}_i = ans(\mathcal{S}_i)$. Let $r(a,b)$ be a fact that is included in $\mathcal{A}_i$. We define the reasoning depth of $r(a,b)$ in $\mathcal{S}_i$, written $depth(r(a,b), \mathcal{S}_i)$, as the minimum number of inference steps that are needed to infer $r(a,b)$ from $\mathcal{S}_i$. For instance, if $r(a,b)$ is included in $\mathcal{S}_i$, we have $depth(r(a,b), \mathcal{S}_i) = 0$. Similarly, if $\mathcal{S}_i$ if a refinement that violates the constraints, we write $depth(\perp, \mathcal{S}_i)$ for the minimum number of inference steps that are needed to establish that the constraints are violated. The *maximum reasoning depth* of a problem instance $(\mathcal{S}, a, b, \mathcal{R})$ is then computed as follows:

$$max\text{-}depth(\mathcal{S}, a, b, \mathcal{R}) = \max\left(\{depth(r(a,b), \mathcal{S}_i) \mid \mathcal{S}_i \in ref^+(\mathcal{S}), r \in \mathcal{R}\}\right.$$
$$\left. \cup \{depth(\perp, \mathcal{S}_i) \mid \mathcal{S}_i \in ref^-(\mathcal{S})\}\right)$$

The reasoning depth is determined by the hardest relation in $\mathcal{R}$ and the hardest answer set.

---

[4]More elaborate examples can be found in the Appendix C.

Reasoning width   Intuitively, the more ambiguity in a given problem instance, the harder it is to solve, all things being equal. We can straightforwardly measure the amount of ambiguity by computing the number of possible refinements of a story $\mathcal{S}$. If each ambiguous fact introduces two possibilities, then the number of possible refinements is 2^N, with N the number of ambiguous facts.[5] However, some ambiguous fact may not play any role in the derivation of the query, so simply counting the number of refinements may be misleading. As an alternative, we therefore focus on counting the number of unique derivations. In particular, we define the reasoning width of a fact $r(a,b)$ w.r.t. a story $\mathcal{S}$ as:

$$width(r(a,b), \mathcal{S}) = |\{proof(r(a,b), \mathcal{S}_i) \,|\, \mathcal{S}_i \in \mathit{ref}^+(\mathcal{S})\}| + |\{proof(\bot, \mathcal{S}_i) \,|\, \mathcal{S}_i \in \mathit{ref}^-(\mathcal{S})\}|$$

where we write $proof(r(a,b), \mathcal{S}_i)$ for the derivation which proves that $r(a,b)$ can be derived from $\mathcal{S}_i$. If there are multiple proofs, we fix $proof(r(a,b), \mathcal{S}_i)$ to be the shortest one, with ties broken arbitrarily. Similarly, $proof(\bot, \mathcal{S}_i)$ denotes a minimal proof that $\mathcal{S}_i$ violates the constraints. In other words, the width of $r(a,b)$ is the sum of the number of distinct derivations of $r(a,b)$, across all the answer sets of $\mathcal{S}$, and the number of distinct derivations of constraint violation, across all refinements of $\mathcal{S}$ without an answer set. The maximal width of a problem instance $(\mathcal{S}, a, b, \mathcal{R})$ is then computed as the reasoning width for the hardest relation in $\mathcal{R}$:

$$max\text{-}width(\mathcal{S}, a, b, \mathcal{R}) = \max\{width(r(a,b), \mathcal{S}) \,|\, r \in \mathcal{R}\}$$

Non-path reasoning   In the case of CLUTRR, all the required rules are of the following form

$$r(X, Z) \text{:-} r_1(X, Y), r_2(Y, Z) \tag{2}$$

If all rules are like this, then the problem of inferring a relational fact $s(a,b)$ boils down to (i) finding an informative path connecting a and b (where we view the facts as the edges of a knowledge graph), and (ii) repeatedly applying rules of the form (2) to replace two adjacent edges by a single edge (representing the composition of the two given relations), until we end up with a single edge connecting a and b.[6] Many approaches for systematic relational reasoning are closely aligned with this idea. As such, problem instances that require going beyond this kind of path-based reasoning can be expected to present difficulties for many models. We introduce two metrics to measure the extent to which a problem instance requires going beyond path-based reasoning.

The first metric, **backtrack load (BL),** is based on the observation that for path-based derivations, the number of inference steps is always one less than the number of entities involved in the derivation. In contrast, for more complex derivations, we often see a higher number of inference steps, relative to the number of entities. We thus define $BL(\tau)$ for a derivation τ as the ratio of the number of inference steps to the number of entities involved. The maximum backtrack load of a problem instance is then:

$$max\text{-}BL(\mathcal{S}, a, b, \mathcal{R}) = \max\{BL(proof(r(a,b), \mathcal{S}_i)) \,|\, \mathcal{S}_i \in \mathit{ref}^+(\mathcal{S}), r \in \mathcal{R}\}$$

The second metric is called **off-path edge count (OPEC)**. For a given derivation τ of a fact $r(a,b)$, we define $OPEC(\tau)$ as the number of edges that appear in τ which are not on any direct path between a and b (if we view relational facts as the edges of a knowledge graph). We then define the maximal OPEC of a problem instance as:

$$max\text{-}OPEC(\mathcal{S}, a, b, \mathcal{R}) = \max\{OPEC(proof(r(a,b), \mathcal{S}_i)) \,|\, \mathcal{S}_i \in \mathit{ref}^+(\mathcal{S}), r \in \mathcal{R}\}$$

We drop the prefix max- and refer to these objects as BL and OPEC. Figure 2 illustrates how OPEC measures the extent of non-path reasoning.

These two metrics are complementary. BL captures whether the reasoning process needs to go back-and-forth along a given relational path. This back-and-forth reasoning is often required for the problems in our benchmark, even when all the edges involved are on a single path between the two query entities. We expect this to be challenging for many approaches, especially methods such as NCRL and R5 which by design only make a single pass over a given path. OPEC captures whether any off-path reasoning is required. Path-based models typically ignore any edges that are not on a direct path between the two query entities. For more discussion see Appendix I.

4.3   Training distribution and held-out test sets

---

[5]This follows because the alternatives occurring in different ambiguous facts never overlap in our dataset.
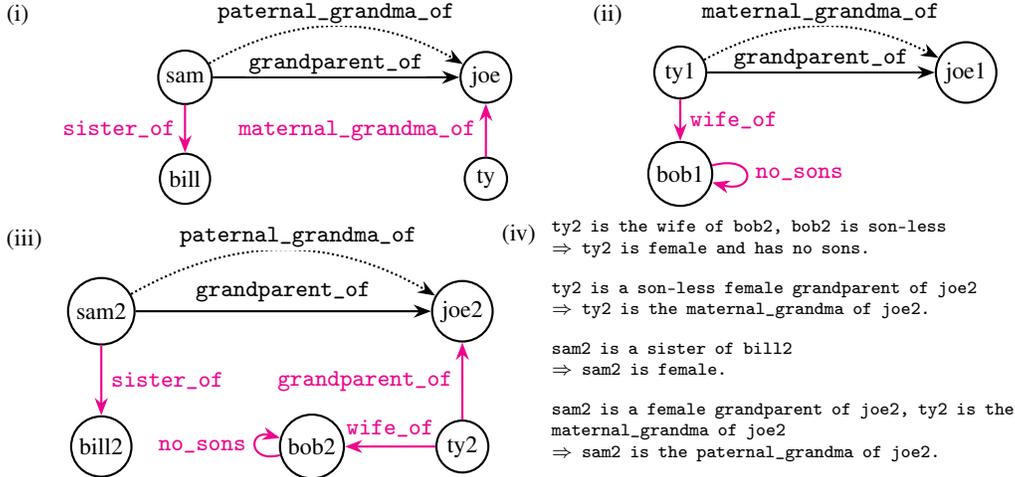[6]See [Khalid and Schockaert, 2025] for a formal proof of this claim.

Figure 2: Source entities are *sam*, *ty1*, and *sam2*, while target entities are *joe*, *joe1*, and *joe2* for the queries accompanying stories (i), (ii) and (iii), respectively. Solid edges represent the relationships explicitly in the story. Dashed edges are entailed relationships between source–target pairs. Pink edges indicate edges that do not lie on any path between the source and target. Panel (iv) illustrates a derivation of the entailed fact in story (iii). It uses all four off-path edges, hence the query from story (iii) has an OPEC value of 4. The queries in stories (i) and (ii) each have an OPEC value of 2.

The training set for NoRA contains examples whose difficulty, according to the four proposed metrics, is controlled: reasoning depth ≤ 6, reasoning width ≤ 5, BL ≤ 1.5 and OPEC ≤ 2. The marginal distribution of these four metrics within the training dataset covers a variety of examples (Appendix 9), which is essential for enabling models to generalize systematically. We have also created a separate training set which is free of ambiguity, i.e. where all examples have reasoning width 1. To rigorously test generalization, we define several held-out evaluation subsets, each focused on specific types of reasoning that go beyond what the model encounters during training. We have four such out-of-distribution test sets involving ambiguities and three which do not. Each of these out-of-distribution test sets extends the difficulty level

Table 1: Overview of the dataset splits. Values that require the model to generalize from the training distributions are highlighted in red.

Name	Depth	Width	BL	OPEC
Train-a	≤ 6	≤ 5	≤ 1.5	≤ 2
Train-na	≤ 6	**1**	≤ 1.5	≤ 2
Test-D	**> 6**	≤ 5	≤ 1.5	≤ 2
Test-W	≤ 6	**> 5**	≤ 1.5	≤ 2
Test-BL	≤ 6	≤ 5	**> 1.5**	-
Test-OPEC	-	-	-	**≥ 3**
Test-In-dist	≤ 6	≤ 5	≤ 1.5	≤ 2
Test-D-na	**> 6**	**1**	≤ 1.5	≤ 2
Test-BL-na	≤ 6	**1**	**> 1.5**	-
Test-OPEC-na	-	**1**	-	**≥ 3**
Test-In-dist-na	≤ 6	**1**	≤ 1.5	≤ 2

of the problem instances according to one of the considered difficulty metrics. Finally, we also created in-distribution test sets, containing unseen problem instances with similar characteristics as those from the training set. An overview of the different datasets is shown in Table 1.

5 Experiments

We evaluate a number of state-of-the-art models on NoRA. Pure path-based methods, such as NCRL and R5, are limited to path-based inference by design, and are thus not suitable. CTPs are too inefficient to handle the large number of rules that needs to be learned for NoRA, and they cannot model constraints. We therefore focus our analysis on the following methods. **Edge Transformers** (ETs) [Bergen et al., 2021] are more versatile than other methods for systematic reasoning, and thus a natural candidate for the more challenging setting presented by NoRA. However, they cannot naturally model multiple relationships between the same entities (i.e. the edge index cannot have degeneracies). We therefore consider two versions of ETs: a vanilla ET, where a single relationship is chosen for each entity pair, arbitrarily, and others are simply ignored (single-edge) and a modified ET

Table 2: Results of state-of-the-art models for systematic reasoning on the NoRA test sets.

| | | Trained with ambiguity | | | | Trained without ambiguity | | | |
		In-dist	D	W	BL	OPEC	In-dist-na	D-na	BL-na	OPEC-na
Exact-match Accuracy	ET (single-edge)	0.885	**0.741**	0.703	0.245	**0.060**	**0.800**	**0.822**	**0.104**	**0.110**
	ET (multi-edge)	**0.900**	0.493	**0.790**	**0.785**	0.037	**0.800**	0.494	0.056	0.077
	RAT (single-edge)	0.721	0.494	0.615	0.234	0.042	0.800	0.493	0.092	0.094
	RAT (multi-edge)	**0.900**	0.676	0.668	0.540	0.028	0.827	0.768	0.023	0.017
	EpiGNN-`min` (margin)	0.334	0.491	0.176	0.000	0.000	0.208	0.485	0.000	0.000
	EpiGNN-`min` (BCE)	0.451	0.665	0.456	0.154	0.005	0.475	0.488	0.008	0.025
	EpiGNN-`mul` (BCE)	0.520	0.604	0.491	0.156	0.009	0.539	0.716	0.027	0.045
	NBFNet (margin)	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	NBFNet (BCE)	0.576	0.531	0.460	0.153	0.009	0.679	0.764	0.012	0.043
	R-GCN	0.347	0.672	0.283	0.051	0.032	0.579	0.740	0.018	0.012

in which the edge embeddings are averaged if there are multiple relationships (multi-edge). We also evaluate transformers with **relation-aware self attention** (RAT) [Shaw et al., 2018], as a precursor to ETs. Next, we evaluate **EpiGNNs** [Khalid and Schockaert, 2025], which are the state-of-the-art on STaR (the only existing benchmark that goes beyond path-based systematic reasoning). We consider two variants: one with the original margin loss and one with a binary cross-entropy loss, with the latter intuitively being more suitable for the multi-label setting. We consider both minimum and multiplication for aggregation. Finally, we evaluate **NBFNet** [Zhu et al., 2021] and **R-GCNs** [Schlichtkrull et al., 2018] as representative GNN models. To evaluate these models, we encode ambiguities in the graph representation of stories using special edges (Appendix J).

Main results   The results are shown in Table 2, in terms of exact-match accuracy (i.e. we measure if the model's prediction of the relation set $\mathcal{R}$ exactly matches the ground truth). Models trained on *Train-a* are evaluated on the test sets with ambiguity, while models trained on *Train-na* are evaluated on the remaining test sets. ETs emerge as the best-performing model. All models perform poorly on OPEC, BL-na and OPEC-na. Surprisingly, for most models, performance on test-W is reasonable. Furthermore, all models perform better on BL than on BL-na, despite the fact that BL was assumed to be harder. Further analysis has shown that models are exploiting shortcuts to solve the majority of ambiguous problems (see Appendix K). The GNN methods all perform poorly on the BL and OPEC test sets, which can be explained by their strong alignment with path-based reasoning. In fact, the GNN models are even performing poorly on the in-distribution test sets, for the same reason. Among the GNN models, EpiGNNs with BCE loss and multiplication-based aggregation perform better. The results also confirm that the margin-based loss is unsuitable for the multi-label setting.

Analysis of ET performance   Figure 3 breaks down the performance of the Edge Transformer on test-D, test-W and test-OPEC. Surprisingly, the performance decline is minimal along the considered difficulty axes. For instance, in the case of Test-D, the results for reasoning depth 12 are almost as good as those for reasoning depth 7, for the vanilla model. Similarly, apart from the dip at depth 7, the multi-edge model performs similarly between depths 8 to 12. However, these results have to be interpreted with caution. Recall that the test problems were obtained by random sampling. Obtaining hard instances in this way is difficult, meaning that we cannot easily test how the model would perform when the reasoning depth is higher than 12 or OPEC is higher than 4, for instance. This is something that we have addressed by introducing a variant of our benchmark, called NoRA v1.1, as explained below. Another consequence of the fact that randomly sampled problem instances are rarely hard relates to the correlations between the difficulty metrics. For instance, a problem with high reasoning depth will typically have low OPEC, and a problem with high OPEC will typically have low reasoning depth. Problems with high reasoning depth may thus be solved well because they are easier in other respects, rather than because the generalization abilities of the model. We analyze this in Figure 4a, where we show the performance for different reasoning depths, *while controlling for both BL and reasoning width*. In this case, we can see a dramatic decline in performance when going from reasoning depth 4 (where the multi-edge ET achieves accuracies above 0.8) to reasoning depth 6 (where the performance varies from around 0.2 to 0.6). Interestingly, in this analysis, the multi-edge variant also clearly outperforms the single-edge variant. This reflects the need for more informationally complete input representations for problems with higher BL.
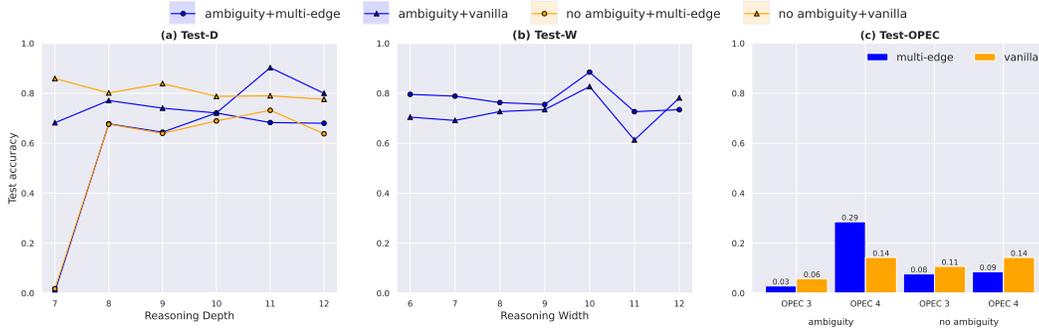
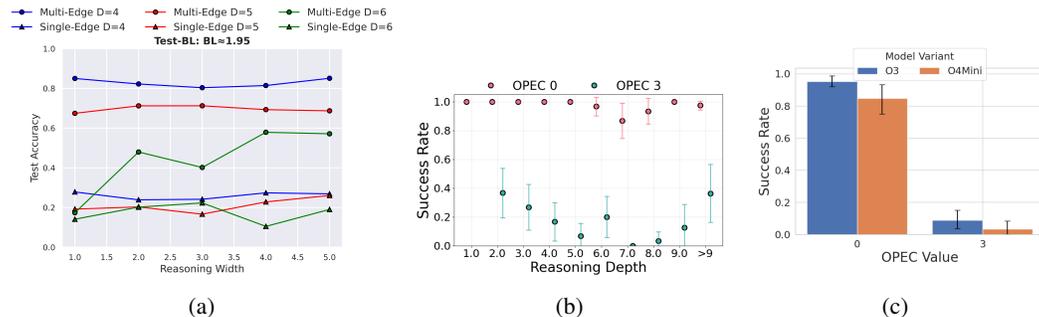Figure 3: Analysis of the performance of ETs on various splits of the dataset.

Figure 4: (a) Breakdown of the performance of edge transformers on Test-D; (b) analysis of o3 on non-ambiguous stories; (c) a comparison between o3 and o4-mini on non-ambiguous stories.

Evaluating Large Reasoning Models   NoRA was designed to test the compositional generalization abilities of neural systematic reasoning models. The defined NoRA world rules are realistic, as most humans and large language models would deem them true or likely (see Appendix E). This is a desirable property for evaluating the systematic generalization and rule learning capabilities of Large Reasoning Models (LRMs) [Zhu et al., 2023]. We also evaluated the LRMs o3 and o4-mini on a subset of NoRA problems, *when explicitly given the entire set of world rules* (only in the LRM experiments are the rules explicitly provided; in all other settings, they must be induced by the model). Being able to apply the correct rules is clearly a prerequisite for solving the considered learning tasks. For this experiment (details in Appendix G), we only consider problem instances without ambiguity, as we want to focus on the extent to which these models can deal with off-path reasoning, and we only consider problem instances where there is a single best label. We provide the models with two in-context demonstrations. Success is measured by exact match with the ground truth label. The results for o3 are shown in Figure 4b. While the model achieves near-perfect accuracy for problem instances with OPEC 0, the performance drops dramatically for OPEC 3, where none of the problem instances of reasoning depth 7 were answered correctly. Even when the world rules are explicitly provided—and are rules the LRM is already familiar with through pretraining—the model fails to apply them correctly to problem instances, highlighting the inherent difficulty of off-path reasoning tasks. Surprisingly, for higher inference depths, the performance is slightly better. This is due to the presence of instances where the LRM can apply shortcuts (Appendix H). As shown in Figure 4c, the performance of o4-mini is slightly worse than that of o3. This non-path-based reasoning analysis aligns with findings from Dziri et al. [2023] on pure reasoning tasks. As an auxiliary task, we tested the model's ability to recover the necessary world rules for solving the task (Appendix G).

Additional datasets: NoRA v1.1 and HetioNet   To further support future work on neural relational reasoning, we introduce two additional datasets. First, we introduce a variant of NoRA, called NoRA v1.1, where problem instances are sampled in a more systematic way, using the recursive subgraph expansion technique [Khalid and Schockaert, 2025]. This has two consequences. First, it means that we can easily create problem instances with larger OPEC, reasoning depth, and BL values. As a result, we can include examples with higher structural complexity in the training set (e.g. allowing OPEC
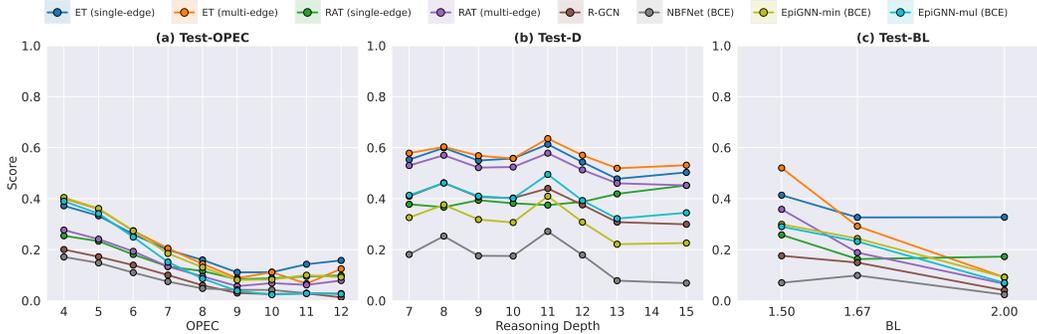
Figure 5: Results for the expanded version of NoRA (v1.1) that uses recursive subgraph expansion to generate harder splits along the axes: (a) OPEC, (b) Reasoning Depth (c) BL.

values up to 3), and include much harder problem instances in the test sets. Second, by generating the problem instances in this way, we can guarantee that every test example can be obtained by a *stitching together* process of multiple training examples. As a result, we are guaranteed that a model which achieves compositional generalization can solve every test instance. To illustrate this stitching together process, consider Figure 2: panels (i) and (ii) depict training instances with OPEC 2. These are combined by (a) deleting the fact `maternal_grandma_of(ty, joe)` from story (i), (b) renaming entities in story (ii) to align with those in story (i) (joe1-> joe etc), and (c) adding the story facts from (ii) to (i). Finally renaming all entities, we obtain story (iii), which is a problem instance with OPEC 4 and is included in the test set. For NoRA v1.1 we did not include any problems with ambiguity. Figure 5 shows the performance of models on NoRA v1.1. The main conclusions from Table 2 remain valid. This demonstrates that the inability of models to handle off-path reasoning remains robust to variations in how the problem instances are generated.

We also introduce another dataset, called **HetioNet**, which was inspired by Himmelstein et al. [2023]. This dataset is based on a completely different set of world rules, unrelated to family relationships. Here, entities correspond to *diseases*, *genes*, and *drugs*, and relations capture biological phenomena. Moreover, the kinds of regularities that models are expected to learn are rather different. For instance, while families are organized into hierarchical structures, no such structure exists in the case of HetioNet. A detailed analysis of HetioNet is provided in Appendix B.2. It shows that, even after this shift in relational regularity, most models continue to struggle on tasks that require off-path reasoning. Surprisingly, however, the EpiGNN outperforms ETs on the test-OPEC split of HetioNet. Further work is needed to better understand the kinds of regularities that different models are able to capture.

6 Conclusions

We have introduced a new benchmark for systematic relational reasoning, called NoRA. It has three core features which makes it more challenging than existing benchmarks: the need for off-path reasoning, the presence of ambiguities, and the fact that entities can be simultaneously related in different ways. We found all methods to struggle significantly with off-path reasoning, suggesting that fundamentally different architectures may be needed to push forward the state-of-the-art in neural relational reasoning. Interestingly, Large Reasoning Models such as o3 were not able to solve problem instances that require off-path reasoning either, even when explicitly given all the required rules. Surprisingly, the presence of ambiguity did not pose any particular challenges for the tested models. However, further analysis revealed this to be due to the presence of shortcuts, allowing models to solve these problem instances without actually needing to reason about ambiguity. This highlights the challenge in generating hard problem instances. Finally, to test the robustness of our findings, we introduced two additional datasets: NoRA v1.1 and HetioNet.

Limitations The ability to measure the difficulty of problem instances is important for testing models in a systematic way. However, metrics such as *reasoning depth* and BL are sensitive to the way in which the knowledge base has been encoded. In the experiments with o3 we saw examples where the "true" reasoning depth was lower than that measured by the metric. All ambiguities are not equally challenging, which is something that *reasoning width* only partially captures.

References

Samy Badreddine, Artur S. d'Avila Garcez, Luciano Serafini, and Michael Spranger. Logic tensor networks. *Artif. Intell.*, 303:103649, 2022. doi: 10.1016/J.ARTINT.2021.103649. URL `https://doi.org/10.1016/j.artint.2021.103649`.

Leon Bergen, Timothy J. O'Donnell, and Dzmitry Bahdanau. Systematic generalization with edge transformers. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 1390–1402, 2021. URL `https://proceedings.neurips.cc/paper/2021/hash/0a4dc6dae338c9cb08947c07581f77a2-Abstract.html`.

Kewei Cheng, Nesreen K. Ahmed, and Yizhou Sun. Neural compositional rule learning for knowledge graph reasoning. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023a. URL `https://openreview.net/pdf?id=F8VKQyDgRVj`.

Kewei Cheng, Nesreen K. Ahmed, and Yizhou Sun. Neural compositional rule learning for knowledge graph reasoning. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023b. URL `https://openreview.net/forum?id=F8VKQyDgRVj`.

William W. Cohen. Graphlog: A benchmark for logical learning on graphs. In *Proceedings of the 2019 International Symposium on Inductive Logic Programming*, 2019.

Andrew Cropper, Sebastijan Dumancic, Richard Evans, and Stephen H. Muggleton. Inductive logic programming at 30. *Mach. Learn.*, 111(1):147–172, 2022. doi: 10.1007/S10994-021-06089-1. URL `https://doi.org/10.1007/s10994-021-06089-1`.

Anirban Das, Manfred Denker, Anna Levina, and Lucia Tabacu. A monte carlo algorithm for multiple stochastic integrals of stable processes. *Stochastics & Dynamics*, 22(8), 2022.

Honghua Dong, Jiayuan Mao, Tian Lin, Chong Wang, Lihong Li, and Denny Zhou. Neural logic machines. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL `https://openreview.net/forum?id=B1xY-hRctX`.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, et al. Faith and fate: Limits of transformers on compositionality. *Advances in Neural Information Processing Systems*, 36:70293–70332, 2023.

Richard Evans and Edward Grefenstette. Learning explanatory rules from noisy data. *J. Artif. Intell. Res.*, 61:1–64, 2018. doi: 10.1613/JAIR.5714. URL `https://doi.org/10.1613/jair.5714`.

Martin Gebser, Benjamin Kaufmann, Roland Kaminski, Max Ostrowski, Torsten Schaub, and Marius Schneider. Potassco: The potsdam answer set solving collection. *Ai Communications*, 24(2):107–124, 2011.

Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In Robert A. Kowalski and Kenneth A. Bowen, editors, *Logic Programming, Proceedings of the Fifth International Conference and Symposium, Seattle, Washington, USA, August 15-19, 1988 (2 Volumes)*, pages 1070–1080. MIT Press, 1988.

Daniel S Himmelstein, Michael Zietz, Vincent Rubinetti, Kyle Kloster, Benjamin J Heil, Faisal Alquaddoomi, Dongbo Hu, David N Nicholson, Yun Hao, Blair D Sullivan, et al. Hetnet connectivity search provides rapid insights into how biomedical entities are related. *GigaScience*, 12: giad047, 2023.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795, 2020.

Muhammad Khalid and Steven Schockaert. Systematic relational reasoning with epistemic graph neural networks. *The Thirteenth International Conference on Learning Representations*, 2025.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2017.

Anna Levina and Viola Priesemann. Subsampling scaling. *Nature communications*, 8(1):15140, 2017.

Vladimir Lifschitz. Twelve definitions of a stable model. In *International Conference on Logic Programming*, pages 37–51. Springer, 2008.

Shengyao Lu, Bang Liu, Keith G. Mills, Shangling Jui, and Di Niu. R5: rule discovery with reinforced and recurrent relational reasoning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL `https://openreview.net/forum?id=2eXhNpHeW6E`.

Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Neural probabilistic logic programming in deepproblog. *Artif. Intell.*, 298:103504, 2021. doi: 10.1016/J.ARTINT.2021.103504. URL `https://doi.org/10.1016/j.artint.2021.103504`.

Pasquale Minervini, Sebastian Riedel, Pontus Stenetorp, Edward Grefenstette, and Tim Rocktäschel. Learning reasoning strategies in end-to-end differentiable proving. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 6938–6949. PMLR, 2020. URL `http://proceedings.mlr.press/v119/minervini20a.html`.

Stephen H. Muggleton and Luc De Raedt. Inductive logic programming: Theory and methods. *J. Log. Program.*, 19/20:629–679, 1994. doi: 10.1016/0743-1066(94)90035-3. URL `https://doi.org/10.1016/0743-1066(94)90035-3`.

Ali Sadeghian, Mohammadreza Armandpour, Patrick Ding, and Daisy Zhe Wang. DRUM: end-to-end differentiable rule mining on knowledge graphs. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 15321–15331, 2019. URL `https://proceedings.neurips.cc/paper/2019/hash/0c72cb7ee1512f800abe27823a792d03-Abstract.html`.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In Aldo Gangemi, Roberto Navigli, Maria-Esther Vidal, Pascal Hitzler, Raphaël Troncy, Laura Hollink, Anna Tordai, and Mehwish Alam, editors, *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer, 2018. doi: 10.1007/978-3-319-93417-4_38. URL `https://doi.org/10.1007/978-3-319-93417-4_38`.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 464–468. Association for Computational Linguistics, 2018. doi: 10.18653/V1/N18-2074. URL `https://doi.org/10.18653/v1/n18-2074`.

Koustuv Sinha, Shagun Sodhani, Jin Dong, Joelle Pineau, and William L. Hamilton. Clutrr: A diagnostic benchmark for inductive reasoning from text. In *Proceedings of EMNLP-IJCNLP*, pages 4505–4514. Association for Computational Linguistics, 2019. doi: 10.18653/V1/D19-1458. URL `https://doi.org/10.18653/v1/D19-1458`.

Gustav Sourek, Vojtech Aschenbrenner, Filip Zelezný, Steven Schockaert, and Ondrej Kuzelka. Lifted relational neural networks: Efficient learning of latent relational structures. *J. Artif. Intell. Res.*, 62:69–100, 2018. doi: 10.1613/JAIR.1.11203. URL `https://doi.org/10.1613/jair.1.11203`.

Wangtao Sun, Shizhu He, Jun Zhao, and Kang Liu. From chain to tree: Refining chain-like rules into tree-like rules on knowledge graphs. *CoRR*, abs/2403.05130, 2024. doi: 10.48550/ARXIV.2403. 05130. URL `https://doi.org/10.48550/arXiv.2403.05130`.

Fan Yang, Zhilin Yang, and William W. Cohen. Differentiable learning of logical rules for knowledge base reasoning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 2319–2328, 2017. URL `https://proceedings.neurips.cc/paper/2017/hash/0e55666a4ad822e0e34299df3591d979-Abstract.html`.

Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal Xhonneux, and Jian Tang. Neural Bellman-Ford networks: A general graph neural network framework for link prediction. *Advances in neural information processing systems*, 34:29476–29490, 2021.

Zhaocheng Zhu, Yuan Xue, Xinyun Chen, Denny Zhou, Jian Tang, Dale Schuurmans, and Hanjun Dai. Large language models can learn rules. *arXiv preprint arXiv:2310.07064*, 2023.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The paper and the abstract are aligned in content and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: There is a paragraph discussing the limitations at the end of the conclusions section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof? [NA]

   Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The details of the dataset generation process, as well as the complete description of our evaluation methodology, are provided in the appendix materials. The training and held-out test datasets are publicly accessible via Hugging Face. As stated during the initial submission, both the dataset URL and the corresponding Croissant metadata file were included. The code used for data generation was also shared at that time and includes straightforward execution instructions. All datasets, along with the code used to generate them, are publicly available. In addition, the code for training and evaluating the models is hosted in a public repository. Links to all these resources are provided directly within the paper.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The dataset has been shared. The code used for generating and evaluating the dataset was also shared, with instructions for creating environment and running the python code.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Broad details are provided in the main paper. The full details are included in Appendix. Code is available publicly.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: This information is in the supplementary materials.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: specifics are included in Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The research conducted in the paper fully conforms with the Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: We foresee no immediate scope for potential malicious or unintended uses, fairness considerations, privacy considerations, and security considerations.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: No pretrained language models, image generators, or scraped datasets are created.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: **Clingo** (version 5.7.1): We used the Clingo ASP solver [Gebser et al., 2011], available at `https://potassco.org/clingo/`. The source code is available at `https://github.com/potassco/clingo` and the Python package at `https://pypi.org/project/clingo/`. Clingo is distributed under the MIT License.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The datasets corresponding to the three reasoning worlds—NORA, NORA-1.1, and INSPIREDFROMHETIONET—are hosted publicly on Hugging Face. Each dataset is accompanied by a validated Croissant metadata file, following the NeurIPS 2025 Datasets and Benchmarks guidelines (`https://neurips.cc/Conferences/2025/DatasetsBenchmarks-FAQ`). The three Croissant files were individually validated, packaged into a single ZIP archive, and uploaded as required. A central Hugging Face landing page provides unified access to all three datasets. In addition, the code used to generate the datasets is openly available in a public GitHub repository.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification:[NA]

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Code and Resources

We maintain a YouTube channel containing three lecture videos that provide an overview of the paper, along with an additional tutorial video demonstrating how to download the dataset and begin training reasoning models with minimal background knowledge: `https://youtube.com/playlist?list=PLLC2I1xkrX1D4vxLelWpsJwvpePK7rsxh&si=ykXraXYfopOmL8fr`

The codebase used for generating examples with ASP (Answer Set Programming) is publicly available at: `https://github.com/axd353/WhenNoPathsLeadToRome.git`. The code for conducting experiments with models such as ET, RAT, and EPIGNN—used to produce the results in Table 2—is available at: `https://github.com/erg0dic/whennopathsleadtorome`.

The datasets corresponding to the three reasoning worlds (NoRA, NoRA-1.1 and INSPIRED-FROMHETIONET) can be accessed collectively at: `https://huggingface.co/datasets/axd353/When-No-Paths-Lead-to-Rome`.

For reference, the complete world-rule specifications for each of these worlds are provided at: `https://github.com/axd353/WhenNoPathsLeadToRome/tree/main/ExplicitWorldRuleFilesForReference`.

B Additional experimental results

B.1 Main results

In the main paper, we reported results in terms of exact match (Table 2). In Table 3, we complement this analysis by reporting the results in terms of weighted F1. The weighted F1-score is calculated as the macro F1-score for each label, aggregated using a weighted mean (based on their frequency in the dataset). Exact-match accuracy requires models to predict all labels correctly when multiple labels are true. The weighted F1 metric still provides positive contribution when at least some labels are predicted correctly accounting for class imbalances. Consequently, this metric can often yield higher scores. This is evident, for instance, in the test-OPEC dataset, where multiple target relations have to be predicted. For example, if the target relations are "aunt" and "maternal aunt", it may be the case that we only need off-path reasoning for predicting "maternal aunt". A model that is not capable of off-path reasoning but that can correctly predict "aunt" would thus still be partially rewarded.

Table 3: Results of state-of-the-art models for systematic reasoning on the NoRA test sets.

| | | Trained with ambiguity | | | | Trained without ambiguity | | |
		D	W	BL	OPEC	D-na	BL-na	OPEC-na
	ET (single-edge)	**0.740**	0.814	0.432	0.413	**0.816**	**0.233**	**0.410**
	ET (multi-edge)	0.335	**0.860**	**0.888**	**0.504**	0.336	0.120	0.394
	RAT (single-edge)	0.329	0.744	0.437	0.399	0.333	0.188	0.383
	RAT (multi-edge)	0.677	0.766	0.747	0.457	0.759	0.067	0.294
Weighted F1	EpiGNN-`min` (margin)	0.326	0.296	0.082	0.116	0.326	0.112	0.206
	EpiGNN-`min` (BCE)	0.625	0.633	0.316	0.180	0.319	0.049	0.218
	EpiGNN-`mul` (BCE)	0.554	0.667	0.320	0.185	0.717	0.076	0.249
	NBFNet (margin)	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	NBFNet (BCE)	0.646	0.665	0.347	0.225	0.775	0.083	0.261
	R-GCN	0.691	0.455	0.189	0.286	0.704	0.122	0.195

B.2 HetioNet

To analyze how the results generalize to other datasets, we present results for another world, in addition to NoRA. This world is called Hetionet and is inspired by Himmelstein et al. [2023].

In the HetioNet world, there are three kinds of entities: compounds, diseases, and genes. Compounds and genes can palliate a disease. Compounds can be used to treat a disease, or they can be marked as unusable for treating a disease. Off-path reasoning emerges because, to be used to treat a disease, a compound must both palliate that disease and have no side effects. Compounds and genes can also upregulate a gene; if an entity upregulates a gene that palliates a disease, then the entity itself palliates that disease. Compounds can be similar to each other in three different ways:

- ss2(c1, c2) means that c1 and c2 palliate the same diseases;

- ss3(c1, c2) means that c1 and c2 either both have side effects or neither has side effects;

- ss1(c1, c2) means that c1 and c2 have the same regulatory properties towards genes.

HetioNet has a significantly different regularity than the NoRA world, particularly because there is no hierarchical tree-like structure—two compounds can be related in multiple ways concurrently (whereas in NoRA, your uncle cannot be your brother). We created training sets, test-D, and test-OPEC-na in a similar way to NoRA. As for NoRA v1.1, all examples observed during testing are stitched-up versions of one or more examples that were seen during training. The data are split as follows: OPEC < 3, BL < 1.33, D < 6 for the train splits and OPEC = 3 for Test-OPEC, D = 7 for Test-D. There are no problem instances with ambiguitiy in the case of HetioNet.

The HetioNet world contains far fewer rules (55) compared to NoRA (284). Moreover, only two to three types of relations can exist between entities of two types—for example, a compound may either upregulate or downregulate a gene. In contrast, numerous types of relationships can hold between two entities that are persons in NoRA. Consequently, HetioNet represents a much simpler and more easily solvable world for most models.

The results for state-of-the-art models are shown in Table 4. In line with the results for NoRA, the edge transformer emerges as the best performing model for the in-distribution set set and the Test-D test set. However, the EpiGNN-`min` model has the best performance on Test-OPEC, presumably due to the strong inductive bias of the min pooling operator in this world.

Table 4: Results of state-of-the-art models on the HetioNet test sets.

	Accuracy (Exact Match)			Weighted F1		
	In dist.	D	OPEC	In dist.	D	OPEC
ET (single-edge)	**0.838**	**0.907**	0.495	**0.936**	**0.958**	0.721
ET (multi-edge)	0.725	0.845	0.486	0.857	0.887	0.680
RAT (single-edge)	0.657	0.756	0.466	0.831	0.811	0.687
RAT (multi-edge)	0.784	0.671	0.641	0.908	0.785	0.768
R-GCN	0.712	0.356	0.541	0.901	0.500	0.847
NBFNet (BCE)	0.742	0.428	0.576	0.879	0.571	0.757
EpiGNN-`min` (BCE)	0.714	0.351	**0.772**	0.837	0.365	**0.863**
EpiGNN-`mul` (BCE)	0.704	0.499	0.624	0.832	0.615	0.812

B.3 In-depth analyses for other baselines

We provide further analysis for a GNN (EpiGNN) in Figure 6 and for RAT in Figure 7, to complement the analysis of edge transformers in the main text. Broadly, the trends observed in the main text hold for other models with respect to length and width generalization. For RAT, the single-edge or vanilla model has a higher OPEC performance than it multi-edge counterpart in Figure 7(c). Also, the multi-edge RAT is better at width generalization in figures 7(d)-(f). We also show a weighted F1 that overcomes class imbalances for some figures which highlights a similar trend to the accuracy curves. For the EpiGNN, the `mul` aggregation function does notably better than `min` for OPEC in figures 6(c) and also on the Test-D split in figures 6(a).

C Notation and task: Intuitive walkthrough

Here we give an intuitive overview using examples instead of formal definitions for the notations introduced formally in the main paper. We focus on stories without ambiguity, as ambiguity is discussed in detail in Appendix J. We use **Answer Set Programming (ASP)** as the underlying language to encode problem instances in NoRA.

The dataset is composed of three parts: **world rules**, **stories**, and **entailed atoms**. The *world rules* define the underlying regularity of relationships in a given universe. These rules are not exposed to the model. The goal of learning models is to infer these hidden rules through example instances and apply them to reasoning tasks. We consider two sets of world rules (i.e. two worlds):
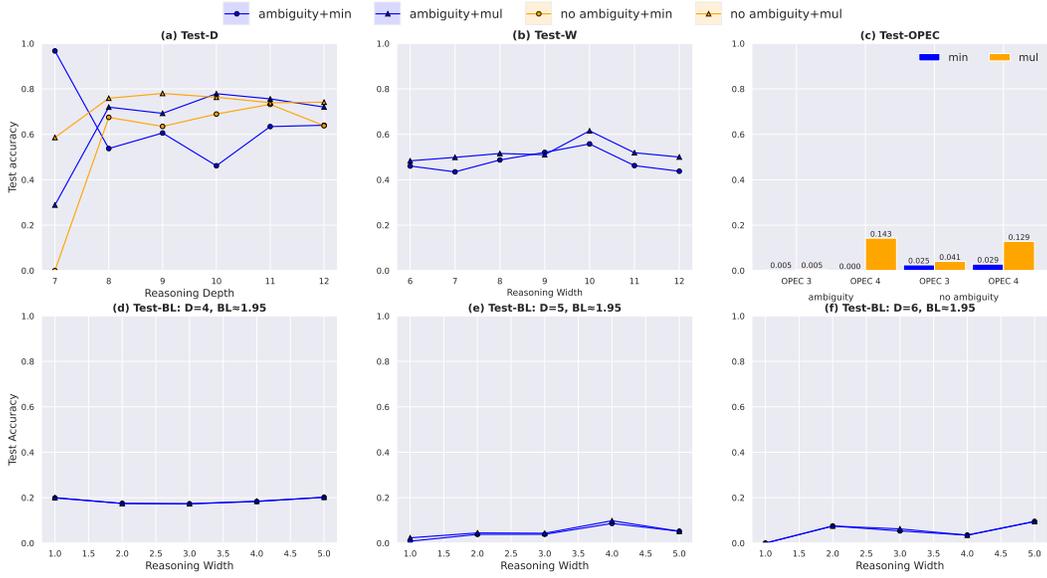
Figure 6: Analysis of the performance of EpiGNN on various splits of the dataset. Weighted F1 scores per class are computed to avoid class imbalances affecting the metric score for the various test splits.
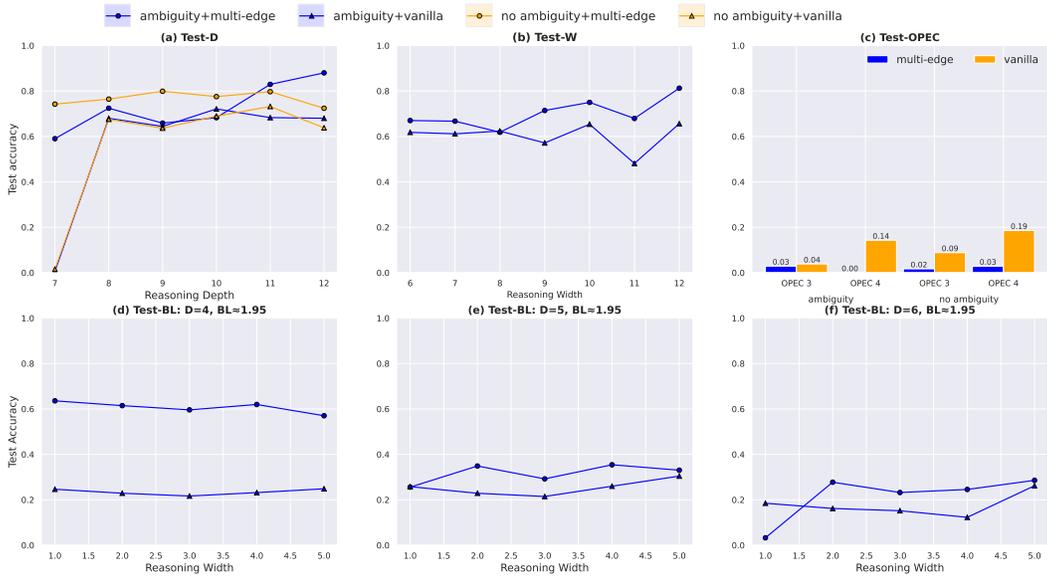

Figure 7: Analysis of the performance of RAT on various splits of the dataset. Weighted F1 scores per class are computed to avoid class imbalances affecting the metric score for the various test splits.

NoRA-mini: A simplified world used for illustrative purposes.

NoRA-full: A richer and more fine-grained world with a broader set of rules, used to generate the full benchmark.

World rules Figure 8(a) shows an example of the world rules in NoRA-mini. These rules fall into three categories: definite rules, constraints and facts.

A **definite rule** consists of a **body** and a **head**. The body is a conjunction of one or more atoms; the head is a single atom. In the absence of constraints, we can think of these rules in terms of standard implication: if all atoms in the body are true, then the head must also be true. For example, consider the following rule from Figure 8(a):

```
living_in_same_place(X,Z) :-living_in_same_place(X,Y),
                                  living_in_same_place(Y,Z).
```

This rule states: for any entities X, Y, and Z, if X lives in the same place as Y, and Y lives in the same place as Z, then X also lives in the same place as Z.

Constraints are rules without a head. They specify sets of atoms that are *not* allowed to be simultaneously true. For example:

```
:- belongs_to(X, underage), parent_of(X, Y).
```

This constraint expresses that an underage person cannot be a parent. Note, in our notation, `rel(X, Y)` means X is `rel` of Y. So, `parent_of(X, Y)` means X is the parent of Y.

Facts are atoms that are always true. They are rules without a body and are often used to declare properties of constants. For example:

```
is_agegroup(underage).
```

Stories Each story consists of a set of *story facts*, which are grounded atoms, i.e., they contain no variables. For example, in Figure 8(b), the fact:

```
school_mates_with(ram, irfan).
```

states that `ram` and `irfan` are schoolmates. Combining the story facts with the fixed world rules one obtains a logic program. Abusing terminology, we sometimes call this logic program the story.

Entailed atoms via stable models Stable models/answer sets are the solution of a logic program. Intuitively, they are a minimal set of atoms/facts that follow from the logic program (see Section N for formal definitions). A stable model includes both the explicitly stated story facts and additional possible atoms that follow logically. These additional atoms are called **entailed atoms**. Figure 8(c) shows the stable model of the story from Figure 8(b). The entailed atoms are highlighted. If an entailed atom has a binary predicate (relationship), its first argument is called the source entity and its second argument the target entity.

Example format and reasoning task While the world rules are kept fixed, multiple logic programs are generated by randomly sampling many sets of story facts. For each such program, the corresponding entailed atoms are computed.

An individual **example** in the dataset consists of:

- The story facts (input), encoded as a graph.
- The target and source entities of an entailed atom, which define the query.

Let a and b be the atoms defined in the query. The task is to predict all relations r such that $r(a, b)$ can be entailed from the story facts. For the example in Figure 8(d), the entailed atom is `living_in_same_place(irfan, lola)`. A model attempting to solve NoRA will be shown

(a) World Rules

Definite Rules

```
living_in_same_place(Y, X) :-
school_mates_with(Y, X).
living_in_same_place(Y, X) :-
belongs_to(X, underage), parent_of(Y,
X).
living_in_same_place(Y, X) :-
living_in_same_place(X, Y).
living_in(Y, Z) :-
living_in_same_place(X, Y), living_in(X,
Z).
belongs_to(X, underage) :-
school_mates_with(X, U).
living_in_same_place(X,Z) :-
living_in_same_place(X,Y),
living_in_same_place(Y,Z).
```

Constraint

```
:- belongs_to(X, underage), parent_of(X,
Y).
```

Facts

```
is_agegroup(underage).
```

(b) Story Facts

```
school_mates_with(ram, irfan).
parent_of(lola, ram).
living_in(irfan, calcutta).
```

(c) Stable Model

```
living_in_same_place(ram, irfan),
living_in_same_place(lola, ram),
living_in_same_place(ram, lola),
living_in_same_place(irfan, ram),
living_in_same_place(lola, irfan),
living_in_same_place(irfan, lola),
living_in_same_place(ram, ram),
living_in_same_place(lola, lola),
living_in_same_place(irfan, irfan),
school_mates_with(ram, irfan),
parent_of(lola, ram),
belongs_to(ram, underage),
living_in(irfan, calcutta),
living_in(ram, calcutta),
living_in(lola, calcutta),
is_agegroup(underage).
```

(d) Visualizing the Reasoning Task
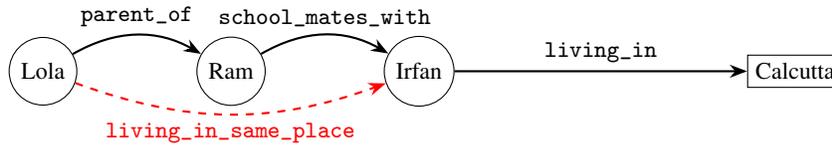
(e) Derivation for `living_in_same_place(irfan, lola)`

```
Fact:  school_mates_with(ram, irfan)
1.  living_in_same_place(ram, irfan) :- school_mates_with(ram, irfan).

2.  living_in_same_place(irfan, ram) :- living_in_same_place(ram, irfan).

3.  belongs_to(ram, underage) :- school_mates_with(ram, irfan).

Fact:  parent_of(lola, ram)
4.  living_in_same_place(lola, ram) :- belongs_to(ram, underage),
        parent_of(lola, ram).

5.  living_in_same_place(ram, lola) :- living_in_same_place(lola, ram).

6.  living_in_same_place(irfan, lola) :- living_in_same_place(irfan, ram),
        living_in_same_place(ram, lola).
```

Figure 8: Illustration of (a) world rules, (b) story facts, (c) stable model with entailed atoms in red, and (d) visual reasoning task: *What is the relationship between Lola and Irfan?* Correct answer: `living_in_same_place`. (e) Reasoning depth for the entailed atom in d.

the story-facts, the source entity `irfan`, and the target entity `lola`, and it must infer all predicates/relationships (including missing ones which is only `living_in_same_place` in this case) between source and target . In NoRA-full, multiple relationships/predicates might be true between the same two entities.

Reasoning depth The difficulty of deriving an entailed atom is influenced by the number of reasoning steps required to reach it, excluding the direct use of story-facts. For example, Figure 8(e) shows that for the given story in NoRA-mini, deriving `living_in_same_place(irfan, lola)` requires six inference steps. Since derivations may not be unique, we use derivations that are minimal (in a sense) to calculate the metric called **reasoning depth**.

D Data generation and sampling

Data generation process   We generate approximately 500,000 example instances by repeatedly sampling random story facts, as detailed below. The same set of world rules is used for all stories (see `https://huggingface.co/datasets/axd353/When-No-Paths-Lead-to-Rome` for the full set of NoRA rules). A single logic program may have multiple entailed atoms, and hence gives rise to multiple example instances in the final dataset. Each story contains two types of entities: *persons* and *places*. First, all entities are generated and assigned a type (person or place). This assignment is governed by a parameter called `person_percent`, which determines the probability that an entity is a person. Higher values of `person_percent` result in more persons, while lower values yield more places. The value of `person_percent` for each story is recorded and included in the dataset. When a person entity is introduced, its gender is either assigned (male or female) or left unspecified. This is controlled by a per-story parameter called `no_gender_assign`, which captures the proportion of person entities with unspecified gender.

Relationships are sampled from the list of binary predicates defined in the world rules . For each story fact, a predicate is sampled and applied to a randomly chosen pair of entities, resulting in a fact of the form `rel(e1,e2)` being added to the story. After each fact is added, the resulting logic program is solved to ensure that at least one answer set exists—i.e., that the story remains consistent. If the added fact introduces a contradiction, it is discarded and a new one is sampled instead. The number of entities per story is sampled uniformly between 20 and 50, and the total number of story facts per instance ranges from 30 to 75. Details of how ambiguous facts are introduced into the stories are provided in Section J.

While there are many possible relationships that can hold between two people, we only consider one relationship between people and places, namely `living_in`. We thus need to make sure that queries where the source entity is a person and the target entity is a place are not trivial to answer, i.e. that models cannot rely on the shortcut that in such cases the answer is always the singleton {`living_in`}. To this end, we have introduced an additional predicate `not_living_in`, which is inferred by the following rule, encoding the fact that a person can only live in one place:

$$\text{not\_living\_in(X,Z) :- living\_in(X,Y), Y} \neq \text{Z}$$

Dataset construction   From this pool of example instances, we construct training and testing datasets under the constraint that *all target query relationships in the test sets must appear in the training data*. To balance the distribution of problem difficulty in the training set, we use *inverse transform sampling*. A general discussion of the nuances of re-sampling techniques can be found in [Levina and Priesemann, 2017, Das et al., 2022]. We use rejections sampling, enabling stratified sampling via quantile functions to obtain the training set. See discussion below:

Difficulty stratification   Examples are binned by four metrics:

- **Reasoning Depth** (3 bins uniformly covering the range),
- **Reasoning Width** (3 bins uniformly covering the range),
- **Branching Length (BL)** (2 bins uniformly covering the range),
- **OPEC** (2 bins: 0 vs. 1–2).

The sampling process follows a rejection-based strategy, beginning with a large pool of candidate examples and iteratively removing samples to achieve a balanced marginal distribution over difficulty metrics. We aim to balance the dataset along several predefined difficulty axes, denoted as S_{diff}. Each axis in S_{diff} corresponds to a difficulty metric—such as reasoning depth, branching length (BL), or OPEC—and is associated with a specific number of target bins.

Sampling proceeds in multiple passes (up to a maximum of `max_p`), terminating early if a satisfactory balance is achieved. In each pass, the following steps are performed:

1. **Score Initialization:** Initialize a removal score of zero for all examples.
2. **Metric-wise Imbalance Scoring:** For each difficulty metric $p_{\text{dm}} \in S_{\text{diff}}$, bin the dataset into `num_bins[`p_{dm}`]` quantile-based bins. Identify the over-represented bins (i.e., bins whose sample count exceeds the target). For every example in an over-represented bin, increment its score by one.

3. **Overrepresentation Removal:** After processing all metrics, make a second pass over S_{diff}. For each over-represented bin, identify examples with the highest accumulated scores and remove them first.

Training Data distributions This two-pass process, repeated across sampling rounds, ensures that examples contributing disproportionately to skewed distributions are pruned while maintaining as much diversity and coverage as possible. Figures 9 show the difficulty metric distributions for Train-A (ambiguous) and Train-NA (non-ambiguous) sets.

(a) Train-A (with ambiguity)

(b) Train-NA (no ambiguity)
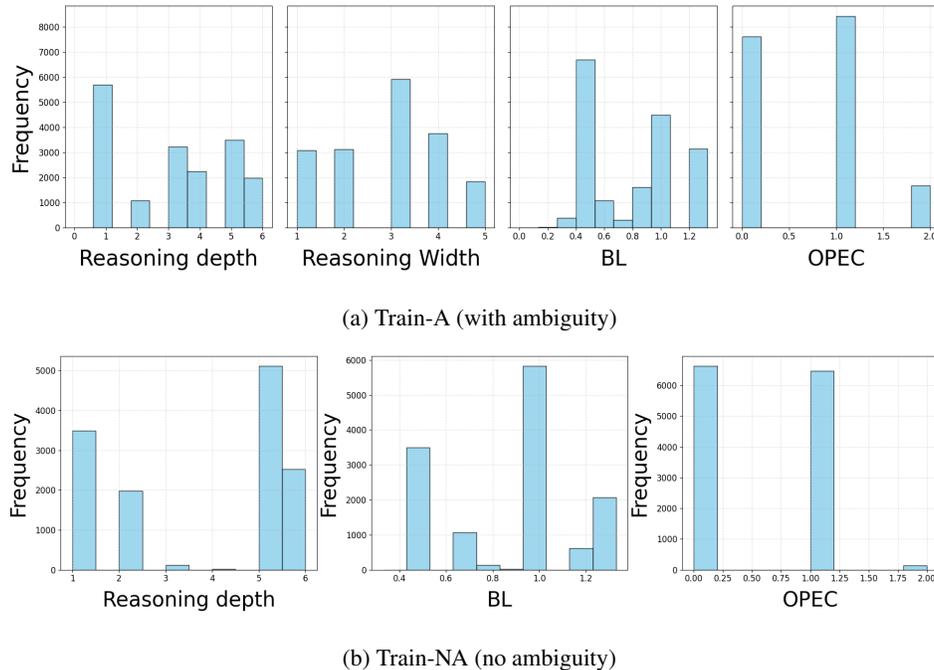
Figure 9: Distributions of difficulty metrics across training sets.

Held-out test data As mentioned in the main text, we evaluate on various held-out test datasets, where each test dataset is designed to be hard according to one difficulty metric while remaining in-distribution (compared to the training set) in terms of other difficulty metrics. For the test datasets with ambiguity:

- For `Test-D`, we ensure that a positive refinement (refinements that have an answer set) has a reasoning depth greater than 6. This is to ensure the problem is actually difficult, as models often take shortcuts by ignoring the derivation of the contradiction in other refinements.
- Likewise, for `Test-BL` and `Test-OPEC`, we make sure a positive refinement has BL > 1.5 and OPEC ≥ 3, respectively.

E NoRA rules reflect real-world intuitions

We contend that the world rules from NoRA are *realistic*, in that human beings are able to intuitively accept them to be true (or at least plausible). We believe this is a useful feature of our dataset, as it makes it easier to compare neural reasoning models, such as the ones we discuss in this paper, with LLM based approaches. To test our hypothesis that the rules are realistic, we used an LLM, namely o4-mini, to complete the 284 rules with zero-shot prompting in an open-ended question answering format. Specifically, given the body of a rule, we asked the model the predict the head.

Here are the results for the three types of NoRA rules:

- Rule type: **constraint**   89.0/90.0 (98.9%) correct

- Rule type: **definite_rule**   184.0/194.0 (94.8%) correct
- Overall accuracy: 96.1%

The prompt we used first defines all predicates:

```
Here are some Predicate Definitions:
- "child_of(X,Y)": "X is a child of Y. Order matters: the first argument is
  the child, the second is the parent"
...
[all predicates are likewise described]
```

The next part of the prompt differs for rules and constraints. For definite rules, where the head is a binary predicate, we use the following prompt:

```
Given that all of the following atoms are true:
grandparent_of(X,Y), belongs_to_group(X, male)

What is the relationship between X, Y?
Provide only the predicate with variables in exactly this format:
rel(X,Y)
What is the predicate name that should replace 'rel'? Your response should
be rel(X,Y), where rel is your guess. If you think multiple predicates
could work, you must choose the most specific one. For example:
- If both brother and sibling are suitable, choose brother as it's more
specific.
```

For a constraint, we instead use the following prompt:

```
Given that all of the following atoms are true:
has_property(Y, no_daughters), daughter_of(X,Y)

Can this combination of facts logically exist?
Answer exactly one of:
[Possible] [Impossible] [Inevitable]
```

As a sanity check, we also tested the LRM (o4-mini) with 90 random constraints not in the NoRA rules, but which use the same predicates. Each of these non-world constraints is a slight modification of NoRA constraints. For example, while ":- aunt_or_uncle_of(Y,X), grandchild_of(Y,X)." is a NoRA constraint stating someone's aunt cannot also be their grandchild, we modified it to the non-world constraint ":- aunt_or_uncle_of(Y,X), grandchild_of(U,V)." This should be possible as U,V and X,Y can be different pairs of people, and thus in the real world there is no obstruction for this to be true. We note the o4-model's response with the same constraint prompts as above for these non-world constraints: the model **always responded with "[Possible]".**

F   Experiments with trainable relational reasoning models

F.1   Loss functions

Margin loss   Let us write $\mathbf{x_i}$ to denote the prediction that is obtained by the model for training example i, and let $\mathbf{r_i}$ denote the embedding of relation r_i. We write $\mathbf{r'_i}$ to denote some negative example, i.e. $\mathbf{r'_i}$ for some $r'_i \in \mathcal{R} \setminus \{r_i\}$, the set of all possible relations in NoRA. In the case of multiple target relation vectors, $\mathbf{r_i}$, we take the average, $\mathbf{\bar{r}_i}$. The overall loss function is:

$$\mathcal{L}_{\text{margin}} = \sum_{i \in \mathcal{D}} \max\left(0, \text{CE}(\mathbf{x_i}, \mathbf{\bar{r}_i}) - \text{CE}(\mathbf{x_i}, \mathbf{r'_i}) + \Delta\right) \tag{3}$$

where CE is the cross entropy function and Δ is the margin value that is set to 1.0 after hyperparameter tuning. The margin loss over multiple models involves an additional sum over the cross entropy differences predicted target relation per model inside the max. At inference time, the target relation is predicted using the negative cross entropy as a score function, with respect to every relation vector in $\mathcal{R}$.
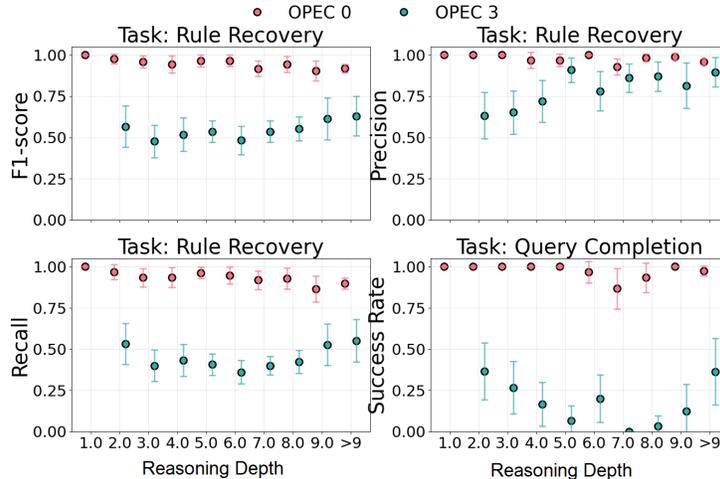
Figure 10: Performance of OpenAI's o3 model on Query Completion and Rule Recovery Tasks. Results separated according to OPEC and the reasoning depth of examples.

Multi-label binary cross entropy   We use a multi-label version of the Binary Cross Entropy (BCE) loss for the multi-label classification setting for all NoRA problems. The logits for each class are transformed using a sigmoid function and then the problem is treated as a binary classification problem with a multi-hot target binary vector.

$$\mathcal{L}_{\text{BCE}} = \sum_{i \in \mathcal{D}, j \in \mathcal{R}} \text{CE}(\sigma(x_{ij}), y_{ij}) \tag{4}$$

where i is the sample index, j is the relation index, x_{ij} is the predicted logit and the y_{ij} is the one-hot target class label.

F.2   Initialization and compute

All trainable parameters for the models are uniformly initialized. All baseline results that were obtained by us were hyperparameter-tuned using grid search, as detailed below. All experiments were conducted using RTX 4090 GPUs. A single experiment using the trainable models can be conducted within a few minutes to 1 hour on a single GPU. This includes training and testing a single model on any test split of NoRA. A single hyperparameter set evaluation is done on about 20% of the total epochs and training data compared to a full experiment and would take a commensurate amount of time.

F.3   Hyperparameter settings

We use the Adam optimizer [Kingma and Ba, 2017]. All the models were hyperparameter tuned using an economical grid search over key parameters. For ET and RAT, a grid search was performed over the number of attention heads, hidden dimension size, the number of message passing rounds, and dropout rate. For the GNNs, we grid searched over the hidden dimension size, the number of message passing rounds. In addition for EpiGNN, we also tuned the number of facets. All the optimal hyperparameters are available in the companion code with the manuscript.

G   Experiments with large reasoning models

Rule recovery task   In addition to the results presented in Section 5 of the main paper on the performance of Large Reasoning Models (LRMs), we evaluate LRM models on a second diagnostic task. Since all NoRA world rules are provided to the model, we additionally task the LRM with outputting the complete set of world rules it used to solve the given query completion task. We call this the *Rule Recovery Task*. Successful completion of the query completion task *without* correct

rule recovery indicates that the model may be taking shortcuts to arrive at the correct answer without following the intended reasoning steps.

Figure 10 presents our results side-by-side with the query completion results (a copy of Figure 4b from the main paper), for easy comparison. This parallel presentation is particularly informative as both tasks are evaluated on identical example instances. The results reveal that while models may have good precision on the rule recovery task, recalling all applicable rules proves substantially more difficult, especially in cases requiring reasoning with significant off-path complexity. The models are evaluated on examples such as those in Figure 4b of the main paper.

For Figure 4b in the main paper, the mean success rate and its 95% confidence interval are estimated using bootstrapping. For Figure 4, the performance of the o3 variant is assessed across different reasoning depths. Mean success rates are computed as sample averages, with confidence intervals derived via normal approximation using standard deviation estimates from a Binomial parameterization.

Prompt format for query completion and rule recovery tasks

The large reasoning model (LRM) is prompted with the following structure for both the query completion and rule recovery tasks:

Section 1: Predicate definitions   *Here are some Predicate Definitions:*

- `grandparent_of(X,Y)`: X is a grandparent of Y. Order matters: the first argument is the grandparent, the second is the grandchild.
- `...` [Additional predicate definitions follow in the actual prompt]

Section 2: World rules   There are three types of rules:

- **A. Definite Rule:** Has a head and a body. It means if all atoms in the body are true, then the head is true.
- **B. Constraint:** Has only a body. It states that the atoms in the body cannot all be true at the same time.
- **C. Fact:** Has only a head. This atom is always true.

Variables are capitalized and rules with variables hold universally for all substitutions.

Here are the NoRA world rules. Rules are indexed and follow the format:

```
Head :- Body.
```

Example:

```
1:  grandparent_of(Y,X) :- grandchild_of(X,Y).
```

[All world rules are then enumerated by index.]

Section 3: Two exemplars   **TASK:** You will be given a story made up of predicates describing relationships between entities ...

Example 1:

```
0 is a is_person.
0 is a is_female.
1 is a is_place.
2 is a is_person.
3 is a is_person.
... [more story facts]
```

Query: What is the relation between 11 and 23? What are the indexes of the world rules you will need to derive this?

Response:

- **query_label:** `niece_of`
- **rules_used:** {192, 64, 194, 46, 23}
- **reasoning:**
  From story fact 23, we know that individual 23 is the maternal aunt of individual 11. Applying world rule 192, we deduce that 23 is the maternal aunt or uncle of 11. Rule 194 generalizes this to `aunt_or_uncle_of`. Rule 23 inverts this relation to yield that 11 is a `nibling` of 23. The story also indicates that 37 is the parent of 11 and has no sons. Applying rule 64, we infer that 11 is female. Rule 46 finally allows us to conclude that 11 is the `niece` of 23.

Example 2:

[Another guided exemplar with similar format]

Section 4: Actual problem instance STORY:

0 is a is_person.
1 is a is_person.
1 is a is_male.
2 is a is_person.
... [More story facts]

QUERY:

What is the predicate between 35 and 6? If a relationship between 35 and 6 is explicitly given in the story facts, and there is some other relationship that is also true, you need to uncover the unstated predicate. If multiple predicates capture the relationship between 35 and 6, choose the most specific one.

What are the indexes of the world rules you will need to derive this?

Expected Output:

- **query_label:** . . .
- **rules_used:** { . . . }
- **reasoning:** . . .

H   Large reasoning models use shortcuts

In the NoRA world rules, the knowledge that "a sibling of my sibling is also my sibling" is *not* explicitly encoded as a definite rule. To prove it, one has to chain through the parent–child relations, repeatedly applying the following three world rules:

(W1) `child_of(Y,X) :- child_of(Z_1,X), sibling_of(Y,Z_1).`

(W2) `parent_of(Y,X) :- sibling_of(Z_1,X), parent_of(Y,Z_1).`

(W3) `sibling_of(Y,X) :- parent_of(Z_1,X), child_of(Y,Z_1), Y ≠ X.`

Even before normalising gendered relations, establishing sibling transitivity therefore demands at least four inference steps. In contrast, LRMs have internalized the following direct rule:

(S) `sibling_of(X,Z) :- sibling_of(X,Y), sibling_of(Y,Z).`

Rule (S) is not a NoRA world rule, yet LRMs (like o3) can apply it, collapsing a multi-hop proof into a single step. Consequently, these tasks that need high reasoning depth are effectively much shallower for such models. Every test instance that o3 solved at a reasoning depth > 9 contained sibling transitivity as a sub-problem, so the model's actual reasoning depth was far lower than our theoretical estimate. An example instance with OPEC > 9 that the o3 model predicts correctly is shown in 11.
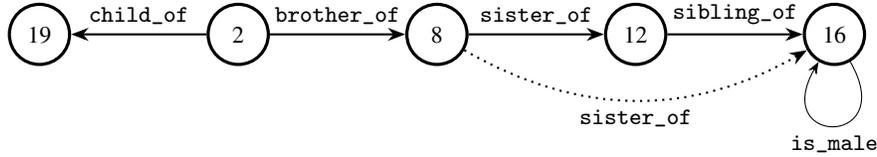
Figure 11: Illustrative fragment of the NoRA graph. Solid edges follow world rules (W1–W3); the dotted edge shows the shortcut (S) inferred by the LRM.

I   Comparing BL and OPEC as measures of non-path reasoning

The Backtrack Load (BL) is the ratio of the number of inference steps to the number of entities involved. As noted in Section M, the number of derivation steps is dependent on the way the world rules are set up. Since we have avoided including redundant rules when specifying the world rules, many problems have a large number of derivation steps. BL is therefore susceptible to overestimating non-path difficulty. An important advantage of BL, however, is that it is capable of identifying non-path reasoning even in cases without off-path edges. On the other hand, OPEC can only identify non-path reasoning when there are off-path edges (i.e. edges which are not on any path between source and target), but it is not dependent on how the world rules are encoded.

BL and OPEC can be controlled independently. For the Test-D, Test-OPEC, and training datasets (as mentioned in the paper), we explicitly control these difficulty metrics to take values within certain limits. To investigate the true correlation between these two difficulty metrics, we explore the stories generated by ASP before sampling to curate datasets.

For the dataset with ambiguity, we observe a Pearson correlation coefficient between OPEC and BL of 0.321 (95% confidence interval via bootstrap: [0.3086, 0.3359]). For the dataset without ambiguity, we observe a Pearson correlation coefficient between OPEC and BL of 0.4650 (95% confidence interval via bootstrap: [0.4503, 0.4840]). Figure 12 breaks down this data.
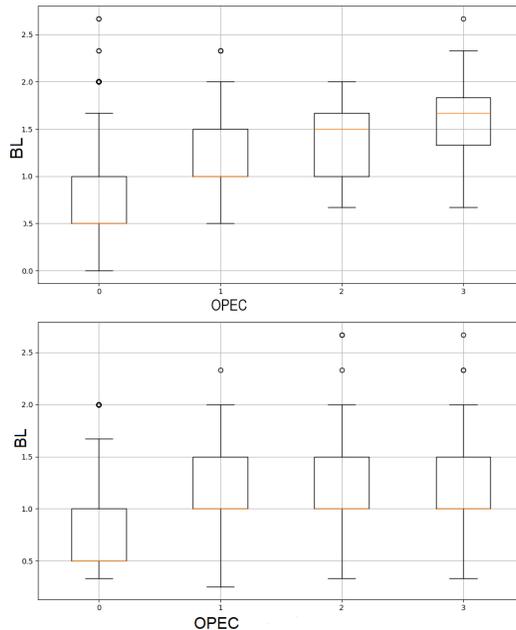

Figure 12: Illustration of the correlation between OPEC and BL using box plots of BL distributions for various OPEC values. The top panel shows data generated **with ambiguous facts**, and the bottom panel shows data generated **without ambiguous facts**.

(a) Ambiguous Story Facts

```
belongs_to(ryan, underage).
school_mates_with(cole, will).
living_in_same_place(sheila, lalit).
living_in(lalit, kgp).
living_in(phil, kgp).
```

```
1{living_in(cole, east_rock);
   living_in(cole, dwight)}1.
1{child_of(ryan, brutus);
   child_of(ryan, cole)}1.
1{colleague_of(brutus, phil);
   colleague_of(brutus, sheila)}1.
```

(b) Refinements and Derivations

(i)
```
colleague_of(brutus, phil),
child_of(ryan, brutus),
living_in(cole, east_rock)
```
```
living_in(brutus, kgp) :-
colleague_of(brutus, phil),
living_in(phil, kgp).

living_in(ryan, kgp) :-
belongs_to(ryan, underage),
parent_of(brutus, ryan),
living_in(brutus, kgp).
```

(ii)
```
colleague_of(brutus, phil),
child_of(ryan, brutus),
living_in(cole, dwight)
```
Same derivation as (i).

(iii)
```
colleague_of(brutus, sheila),
child_of(ryan, brutus),
living_in(cole, east_rock)
```
```
living_in(sheila, kgp) :-
living_in_same_place(sheila, lalit),
living_in(lalit, kgp).

living_in(brutus, kgp) :-
colleague_of(brutus, sheila),
living_in(sheila, kgp).

living_in(ryan, kgp) :-
belongs_to(ryan, underage),
parent_of(brutus, ryan),
living_in(brutus, kgp).
```

(iv)
```
colleague_of(brutus, sheila),
child_of(ryan, brutus),
living_in(cole, dwight)
```
Same derivation as (iii).

(v)
```
child_of(ryan, cole), living_in(cole,
east_rock),
colleague_of(brutus, sheila)
```
```
belongs_to(cole, underage) :-
school_mates_with(cole, will).
:- belongs_to(cole, underage),
parent_of(cole, ryan).
```
Contradiction.

(vi)
```
child_of(ryan, cole), living_in(cole,
dwight),
colleague_of(brutus, phil)
```
Same contradiction as (v).

(vii)
```
child_of(ryan, cole), living_in(cole,
east_rock),
colleague_of(brutus, phil)
```
Same contradiction as (v).

(viii)
```
child_of(ryan, cole), living_in(cole,
dwight),
colleague_of(brutus, sheila)
```
Same contradiction as (v).

Figure 13: (a) An ambiguous story in NoRA, with three cardinality-based facts (highlighted in blue). (b) Each numbered box corresponds to a refinement. The top rectangle in each branch highlights the specific choices made for ambiguous facts, and the body shows the derivation of the entailed atom `living_in(ryan, kgp)` or the contradiction that arises.

J Ambiguous facts, story encodings and reasoning width

Real-world text is often ambiguous or incomplete. One motivation for including ambiguity in NoRA is that relation extraction pipelines based on coreference resolution can introduce noise or uncertainty. To reflect such real-world uncertainty, NoRA includes ambiguous story-facts encoded in ASP using *cardinality facts* of the form `l{atom1; atom2; ...; atomk}u`, which indicates that the number of true atoms in the set `{atom1, atom2, ..., atomk}` lies between `l` and `u` (both inclusive).

33

Once such ambiguous facts are introduced into a story, the resulting logic program may admit multiple *stable models*. An **entailed atom** in this setting is defined as an atom that is part of *every* stable model but is not explicitly listed as a story fact. Figure 13(a) shows an ambiguous story in NoRA that contains three ambiguous facts. These yield $2^3 = 8$ possible refinements, of which four result in contradictions, leaving four consistent stable models. A common atom across all four models is `living_in(ryan, kgp)`, which is thus considered an entailed atom and may be used to construct a dataset example instance.

Ambiguity introduces a new notion of difficulty. For the entailed atom `living_in(ryan, kgp)`, Figure 13(b) shows eight refinements (i–viii), of which v–viii lead to contradictions and share the same structure. Among the positive refinements, refinement i and ii yield identical derivations, as do iii and iv. Intuitively, the **reasoning width** of a query is the sum of:

- the number of distinct derivations/proofs that yield the entailed atom across all stable models, and

- the number of distinct derivations/proofs that lead to contradiction in the remaining refinements.

For the example in Figure 13 (with story facts in 2a and the entailed atom `living_in(ryan, kgp)`), this number is 3. A formal definition of **reasoning width** is provided in the main text.

In the stories of NoRA, only specific types of ambiguous facts are used. These follow the ASP cardinality format:

```
l{atom1; atom2; ...; atomk}u
```

where $k \in \{2, 3\}$, and either $u = l = 1$ (meaning *exactly one* atom is true), or $l = 1$ and $u = k$ (meaning *at least one* atom is true). The atoms used in such ambiguous facts are of the following types:

1. `living_in(a, b`$_\mathtt{i}$`)`, where a is a person and b_i are different possible locations. The same a appears in all atoms of the ambiguous fact, i.e., the ambiguity is over which location a lives in.

2. `rel(a, b`$_\mathtt{i}$`)`, where a and b_i are persons, and `rel` is a binary predicate over people (e.g., `grandparent_of`, `sibling_of`). The same a and the same `rel` are used in all atoms of the ambiguous fact, i.e., the ambiguity is over whom a stands in relation to.

Graph encoding of story facts. When story facts are provided to a GNN, they must be converted into directed graphs. For non-ambiguous facts of the form `rel(a,b)`, we follow the standard convention: draw a directed edge from a to b with edge label `rel`. Special entities like `male` in relationships such as `belongs_to_group(sam,male)` are encoded as self-loops (e.g., `sam` $\rightarrow$ `sam` labeled `is_male`), since neural models using these graphs rely solely on edge labels and cannot learn from node labels.

For ambiguous facts, a dedicated **ambiguous node** is introduced to maintain the structure and support model interpretability. Two types of constructions are used:

- For ambiguous facts of the form `1{rel(a,b`$_\mathtt{1}$`); rel(a,b`$_\mathtt{2}$`); ...; rel(a,b`$_\mathtt{k}$`)}k`, where *at least one* relation is true:

  Add an edge from a to a newly created node p_amb_node with label `rel`, and edges from p_amb_node to each b_i labeled `amb, at least 1 is true`.

- For ambiguous facts of the form `1{rel(a,b`$_\mathtt{1}$`); rel(a,b`$_\mathtt{2}$`); ...; rel(a,b`$_\mathtt{k}$`)}1`, where *exactly one* relation is true:

  Add an edge from a to p_amb_node labeled `rel`, and edges from p_amb_node to each b_i labeled `amb, exactly 1 is true`.

Each ambiguous fact introduces exactly one such `p_amb_node`. This design allows GNN-based models to reason over ambiguous structures using only edge labels (see 14).

(a) Story Facts in ASP Syntax

```
1{sibling_of(tim,lisa); sibling_of(tim,aby); sibling_of(tim,fin)}3.
1{living_in(lisa,kgp); living_in(lisa,rome)}1.
living_in(fin,kgp).
belongs_to_group(tim,male).
```
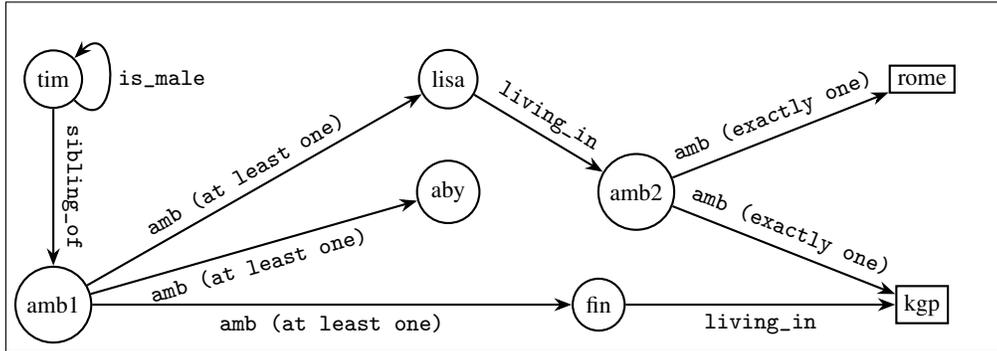
(b) Graph Encoding of Story Facts

Figure 14: (a) An example story in ASP syntax with two ambiguous facts. (b) Corresponding graph encoding: ambiguous facts are handled via auxiliary nodes with labeled ambiguity constraints on edges.

K   Diagnosing model performance on ambiguous stories

We initially expected that handling *ambiguous stories* would pose a significant challenge for the models. However, to our surprise, most models performed nearly as well on the Test-W split as on the training splits. Upon further investigation, we identified that the metric we designed to measure reasoning difficulty—the **reasoning width**—does not fully capture some shortcuts that models can exploit to achieve high performance.

A problem instance (S, a, b, R) with high reasoning width is difficult only if the solver adheres to the ideal reasoning process. However, models can take shortcuts that still very often lead to correct answers. For some instances, these shortcuts fail to yield correct predictions, and in such cases, the performance of the Edge Transformer model deteriorates substantially. Unfortunately, these challenging examples represent only a small fraction of the Test-W dataset.

K.1   Illustrative example of ambiguity

To illustrate this issue, consider the ambiguous story shown in Figure 15. This story contains one ambiguous fact—whether Sean or Shah is the colleague of Rob. This ambiguity gives rise to two refinements.

Query 1: `rob is (brother, sibling) of daisy`

- **Refinement 1:** {`sean is colleague of rob`}
  Derivation: `rob is a male sibling of daisy` → `brother` relationship established.

- **Refinement 2:** {`shah is colleague of rob`}
  Derivation: `shah is underage` → cannot be colleague. Contradiction. Negative refinement.

Query 2: `rob lives in U`

- **Refinement 1:** {`sean is colleague of rob`}
  Derivation: `sean lives in U` → colleague relationship suggests `rob lives in U`.

- **Refinement 2:** {`shah is colleague of rob`}
  Derivation: `shah is underage` → cannot be colleague. Contradiction. Negative refinement.

Figure 16: Two queries derived from the ambiguous story, both with reasoning width 2. The first query can be solved even if the ambiguity is ignored; the second requires handling the ambiguity explicitly.
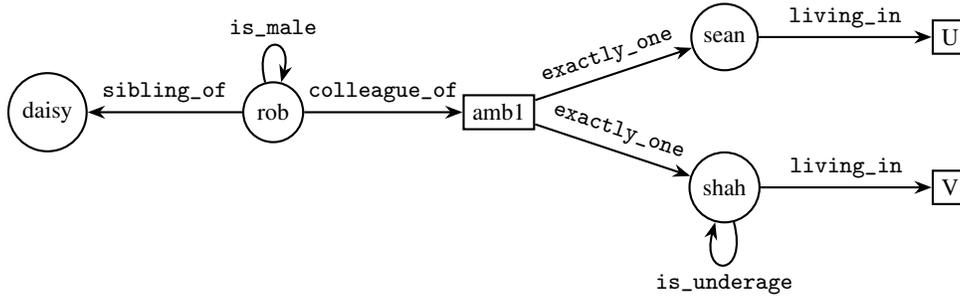

Figure 15: Example ambiguous story containing one ambiguous fact: whether `Sean` or `Shah` is the colleague of `Rob`.

From this story, we derive two entailed facts that can each be turned into problem instances of reasoning width 2, shown in Figure 16. For the first query, an ideal reasoner would identify that the second refinement leads to a contradiction and reason accordingly, yielding two valid proofs—one per refinement—and thus a reasoning width of 2. However, a shortcut reasoner could ignore the ambiguous part of the story and solve the problem without learning and applying the contradiction rule that underage individuals cannot be colleagues. The second query also has reasoning width 2, but here it is essential to apply the contradiction and disprove the second refinement.

K.2 Defining hard ambiguous instances in `Test-W`

To obtain problem instances with ambiguity that are harder to solve, we devise an auxiliary criterion to distinguish the two types of ambiguous problem instances in `Test-W`.

Definition 1 (Hard Ambiguous Problem Instances). A problem instance in `Test-W` is labeled **hard** if:

  (i) An ambiguous fact is used in the derivation of the entailed fact.

  (ii) The entailed fact cannot be derived for all the possible resolutions of the ambiguous fact (i.e. some of the possible resolutions need to be excluded based on the fact that they violate the constraints).

To illustrate condition (ii), consider the story graph in Figure 17.

36

Table 5: Edge Transformer performance on `Test-W` subsets. For comparison, the in-distribution accuracy (same difficulty metric as training) is 90%.

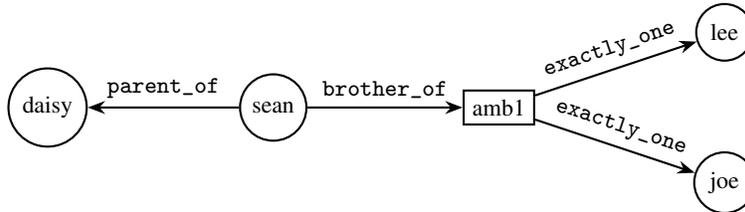|	# Examples	Exact Match Accuracy
Hard Ambiguous Instances	390	51%
Non-Hard	6062	81%

Figure 17: Story graph illustrating an example that satisfies condition (i) but not condition (ii) from Definition 1.

The entailed fact `father_of(sean,daisy)` can be derived as follows:

- **Refinement 1:** From `brother_of(sean,lee)` we derive `is_male(sean)`

- **Refinement 2:** from `brother_of(sean,joe)` we derive `is_male(sean)`

For either resolution of the ambiguous fact we thus obtain `is_male(sean)`. Together with `parent_of(sean,daisy)` we thus derive `father_of(sean,daisy)`.

This instance satisfies condition (i) but not condition (ii) of Definition 1.

K.3 Performance on hard ambiguous instances

The truly challenging examples in `Test-W` (i.e., those satisfying Definition 1) are rare. However, for these examples, the accuracy of the Edge Transformer is substantially lower, as shown in Table 5.

L NoRA v1.1

The world rules for NORA-1.1 are largely the same as those used in NORA. We introduced a few targeted adjustments to address shortcut behaviors identified in Appendix H. In addition, NORA-1.1 does *not* include stories with ambiguity. For reference, the complete world-rule specifications for NORA-1.1, NORA, and INSPIREDFROMHETIONET are available at: `https://github.com/axd353/WhenNoPathsLeadToRome/tree/main/ExplicitWorldRuleFilesForReference`.

The training split and the various test splits for NORA-1.1 are organized as shown in Table 6. Figure 18 shows some basic statistics of the NoRA v1.1 training set.

Table 6: Overview of the NoRA v1.1 dataset splits. Values that require generalization beyond the training distribution are highlighted in red.

Name	Depth	Width	BL	OPEC
Train-na	≤ 6	**1**	< 1.5	≤ 3
Test-D-na	> 6	**1**	< 1.5	≤ 3
Test-BL-na	≤ 6	**1**	≥ 1.5	≤ 3
Test-OPEC-na	$-$	**1**	$-$	> 3
Test-In-dist-na	≤ 6	**1**	< 1.5	≤ 3
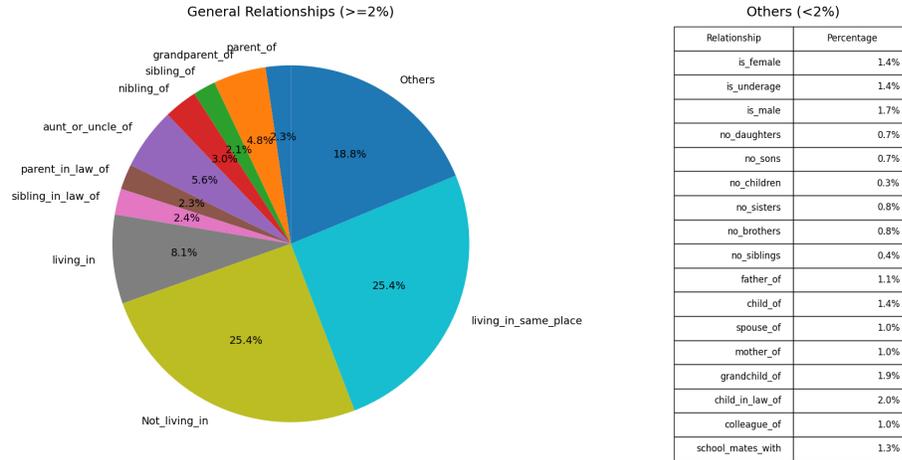
(a) Distribution of predicates/relationships in the NORA-1.1 training set.
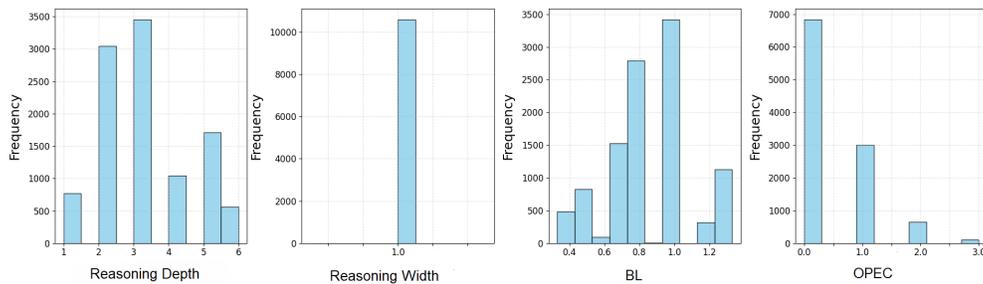
(b) Distribution of difficulty metrics for problem instances in the NORA-1.1 training split.

Figure 18: NORA-1.1 training-set statistics. Top: predicate/relationship frequencies. Bottom: difficulty-metric distribution for training instances.

M Derivation step sensitivity

The number of derivation steps is sensitive to the precise way in which world rules are framed. To illustrate this, consider our NoRA world rules, which are designed to be minimal and avoid redundancy. These rules imply certain relationships (which are not explicit in the world rules). A model could also memorize these implied rules. This would result in shorter derivations but would necessitate memorizing a larger number of rules.

Consider the example in Figure 19. Using the NoRA rules, entailing that Mona is the daughter of Tim requires six derivation steps. However, a rule not explicitly stated in the NoRA world rules, but
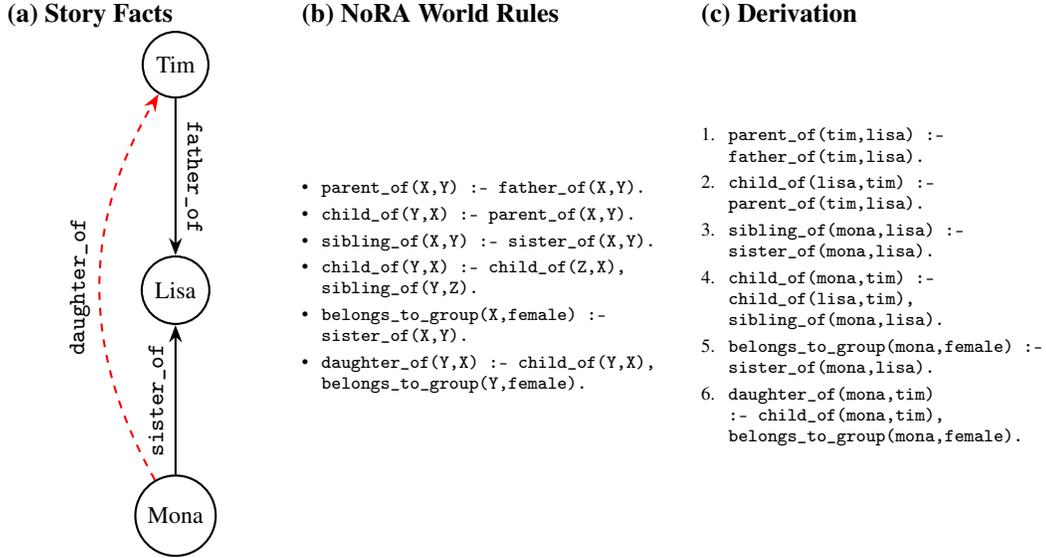
(a) Story Facts

(b) NoRA World Rules

- `parent_of(X,Y) :- father_of(X,Y).`
- `child_of(Y,X) :- parent_of(X,Y).`
- `sibling_of(X,Y) :- sister_of(X,Y).`
- `child_of(Y,X) :- child_of(Z,X), sibling_of(Y,Z).`
- `belongs_to_group(X,female) :- sister_of(X,Y).`
- `daughter_of(Y,X) :- child_of(Y,X), belongs_to_group(Y,female).`

(c) Derivation

1. `parent_of(tim,lisa) :- father_of(tim,lisa).`
2. `child_of(lisa,tim) :- parent_of(tim,lisa).`
3. `sibling_of(mona,lisa) :- sister_of(mona,lisa).`
4. `child_of(mona,tim) :- child_of(lisa,tim), sibling_of(mona,lisa).`
5. `belongs_to_group(mona,female) :- sister_of(mona,lisa).`
6. `daughter_of(mona,tim) :- child_of(mona,tim), belongs_to_group(mona,female).`

Figure 19: Example showing a derivation using a minimal number of rules.

implied by them, is:

$$\texttt{daughter\_of(Z,Y) :- father\_of(X,Y), sister\_of(Z,Y).}$$

If models were to learn such implied rules directly, the derivation for the same entailment would be reduced to a single step.

CLUTRR does not count inverse relationships, such as `parent_of(X,Y) :- child_of(Y,X)`, as derivation steps, whereas such steps are counted in NoRA. Since we have diverse types of rules in NoRA, making a judgment on what counts as a derivation step requires more consideration.

N   Stable models

Solving a logic program involves computing its **stable models**, which are also known as **answer sets** [Lifschitz, 2008]. First note that while we usually specify ASP programs using rules with variables, the semantics of answer sets is defined w.r.t. the grounding of such programs. A ground rule is obtained by replacing the variables in an ASP rule by constants that appear in the program. The grounding of an ASP program consists of all the possible ground rules that we can obtain from its rules. Let us now assume that P is a ground program (i.e. the grounding of an ASP program).

In the absence of rules without negation-as-failure, a stable model of P is a minimal set of atoms, such that:

1. If we assign `true` to every atom in the set, and `false` to all other possible atoms, then all rules in P are satisfied.
2. No strict subset of the model satisfies the above condition.

For rules with negation-as-failure, answer sets are defined in terms of the Gelfond-Lifschitz reduct. While we do not explicitly rely on negation-as-failure in our encoding, for ambiguous facts (see below), we use a language construct that under the hood is translated to such rules. Some of the rules then have conditions with negation-as-failure, of the form *not* $r(a, b)$. Such conditions are intuitively satisfied unless $r(a, b)$ can be inferred. The Gelfond-Lifschitz reduct of a logic program P w.r.t. the answer set A is the logic program P^A that we obtain as follows:

- Any rule with a condition of the form *not* $r(a, b)$ such that $r(a, b) \in A$ is removed from the program.
- Every condition of the form *not* $r(a, b)$ such that $r(a, b) \notin A$ is removed from the body of the rule in which it occurs.

Note that the reduct P^A no longer contains negation-as-failure. We then say that A is an answer set of P iff it is an answer set of the reduct P^A.

Intuitively, a stable model includes both the explicitly stated story facts and additional atoms that follow logically.