A LOGICALLY COMPLETE MODEL FAMILY FOR GRADIENT-BASED RULE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Inductive Logic Programming (ILP) learns logical rules from data, forming an interpretable machine learning model. Neuro-symbolic ILP is an emerging area where logical rules are discovered via gradient-based learning. However, existing models are limited to constrained language biases, hampering further applicability for complex rules. In this work, we propose a neuro-symbolic ILP model family that is logically complete for Horn rules. Our model relaxes all syntactically correct rules into continuous spaces and estimates the gradient to search for the semantically correct solutions. In particular, the gradient involves direct learning of logical variables, which is intractable in existing models. Experiments on standard benchmarks and recently proposed benchmarks with complex rules show that our model outperforms existing methods and LLMs.

1 INTRODUCTION

Inductive Logic Programming (ILP) aims to learn interpretable logic models (Muggleton & Raedt, 1994; Muggleton et al., 2012; Cropper et al., 2020a; Cropper & Dumancic, 2022; Zhang et al., 2024), represented by formal logics, which can generalize uniformly by learning from a small amount of entities. For example, logical formulas in first-order logic generalize to any entity (i.e., constant, object) uniformly, intrinsically endowed with robust generalization ability. In addition, ILP is able to learn from a small amount of data (Evans & Grefenstette, 2018), rendering it highly data-efficient.

ILP originates from symbolic methods (Law et al., 2020; Cropper & Morel, 2021b; Cropper, 2022; Cropper & Hocquette, 2023; Feldstein et al., 2024), which often suffer from the combinatorial explosion problem (Cropper & Morel, 2021a) and from handling noisy data (Hocquette et al., 2024). Neuro-symbolic ILP methods (Evans & Grefenstette, 2018; Gao et al., 2024b), which leverage common techniques from Deep Learning (Bengio et al., 2017), such as continuous parameterization, probabilistic modeling, and gradient-based optimization, show great potential in scaling up. Neuro-symbolic methods surpass symbolic methods in relatively large-scale tasks and noisy settings (Glanois et al., 2022). Consequently, our work focuses on neuro-symbolic ILP.

As an emerging research topic, existing neuro-symbolic ILP methods concentrate on limited language biases (i.e., syntax), impeding larger applications. For instance, existing works focus on learning unary and binary predicates, incapable of capturing hyper-relations involving multiple entities, which are helpful in real-world applications (Guan et al., 2019; Galkin et al., 2020; Wang et al., 2021). In addition, most neuro-symbolic ILP methods limit the number of body atoms to at most 2, relying on difficult predicate invention and requiring more steps at inference. Our work aims to learn Horn rules, a type of rule widely used in ILP, under unconstrained language bias.

Our main contribution is Horn Rule Network (HRN), a neuro-symbolic ILP model family that relaxes all syntactically correct Horn rules into continuous spaces and is optimized via gradient-based optimizers to find semantically correct hypotheses. Significantly different from existing neuro-symbolic ILP, HRN family is not restricted to limited language bias and do not require specific rule templates or candidate rules. HRN enables direct learning of logical variables and atoms, instead of manually assigning variables, which endow with better applicability when the predicates' arity and number of body atoms are larger. The direct learning of variables is not trivial since the forward chaining, the deduction procedure, is not differentiable with respect to logical variables. HRN adopts a stochastic model structure to enable the direct learning of variables.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Our main theoretical and technical innovations are three-fold. First, we prove that HRN family is logically complete for learning Horn rules. Second, to efficiently sample body atoms from HRN's stochastic structures in training, we propose a linear sampling algorithm, which mathematically is a sampling of multiple independent and non-identical distributed Bernoulli variables under an inequality constraint. Third, to estimate the gradient of HRN, we propose a gradient estimator according to HRN's specific stochastic structures. We theoretically prove the estimator's unbiasedness and experimentally verify its effectiveness.

We conduct experiments on standard benchmarks with relatively simple rules and recently proposed tasks with complex rules, which indicate that HRN performs better than existing methods and LLMs.

2 Related work

ILP and logic models has successful applications in knowledge base completion (Qu et al., 2021; Minervini et al., 2020; Lajus et al., 2020; Cheng et al., 2023), bioinformatics (Inoue et al., 2013; Kaalia et al., 2016; Srinivasan et al., 2025), robotics (Sammut et al., 2015; Cropper & Muggleton, 2015; Antanas et al., 2015; Chaghazardi et al., 2024), physics-informed learning (Dai et al., 2017), program analysis (Albarghouthi et al., 2017; Sivaraman et al., 2019; Bartha & Cheney, 2019), ecology (Bohan et al., 2017; Varghese et al., 2022), games (Legras et al., 2018; Cropper et al., 2020b), sequence processing (Evans et al., 2021; 2022; Gao et al., 2025), etc. Advanced research attempts to integrate ILP or logic models with large language models (LLMs) (Yang et al., 2024; Xu et al., 2025) and multimodal large language models (MLLMs) (Peng et al., 2025). In this work, we focus on developing backbone ILP model and extend it to richer language biases.

Neuro-symbolic ILP is an emerging area, which relaxes discrete hypothesis spaces into continuous spaces and optimizes parameters via gradient-based optimizers. Yang et al. (2017) propose a differentiable learning of logical rules, but limited to chain-like rules. Evans & Grefenstette (2018) design a relatively more general *rule-level* relaxation ∂ILP but limited to binary predicates and at most two body atoms. Difflog (Si et al., 2019) leverages another rule-level relaxation and requires a rule-candidate set, also restricted to limited language bias and intractable for large-scale ILP tasks in terms of memory consumption (Chen et al., 2025). Similar rule-level relaxations have been explored in ∂ILP-ST (Shindo et al., 2021), αILP (Shindo et al., 2023), and NEUMANN (Shindo et al., 2024), but require high-quality hypothesis space to learn correct rules (Shindo et al., 2021).

LRI (Campero et al., 2018) proposes a *predicate-level* relaxation, though benefiting from fine-grained continuous modeling, but requires task-specific templates and is thus not extensible. Several logic programming works (Rocktäschel & Riedel, 2017; Maene & Raedt, 2023), though supporting inductive logic programming by predicate-level relaxations, also leverage task-specific templates. To discard the dependence on specific templates, Glanois et al. (2022) provide a theoretically complete set of proto-rules and constructs the HRI model. However, these proto-rules can only be instantiated as Horn rules with unary & binary predicates and with at most two body atoms, i.e., restricted to a constrained language bias. D-LFIT (Gao et al., 2022) and DFORL(Gao et al., 2024a;b) learn same-head Horn programs, i.e., all the rules share the same head atom, but also only consider unary & binary predicates. In contrast to existing works, HRN family covers all sets of Horn rules, bypassing hand-crafted templates or restricted language bias.

3 Preliminaries

3.1 Inductive logic programming

Horn rule is a class of logical rules in first-order logic[1], which is commonly used in artificial intelligence (Lloyd, 2012). Take $head(X_1, X_2) \leftarrow body_1(X_2, X_3, X_1) \wedge body_2(X_1) \wedge body_3(X_3, X_3)$ as an example. The $head, body_1, body_2, body_3$ are *predicates* with arity $2, 3, 1, 2$, respectively, i.e., they can take $2, 3, 1, 2$ arguments, respectively. The X_1, X_2, X_3 are logical *variables*, which are possible values of the arguments. The $head(X_1, X_2)$ is called *head atom*, and the $body_1(X_2, X_3, X_1), body_2(X_1), body_3(X_3, X_3)$ are called *body atoms*. These atoms are called *un-*

[1]We assume basic knowledge of first-order logic. Following previous neuro-symbolic works, we consider function-free Horn rules.

ground atoms since they only have variables and not constants. Its counterpart is *ground atom*, also known as Horn *fact*, which is constituted by a predicate and several *constants*.

ILP learns a hypothesis $\mathcal{H}$, a set of Horn rules, from given background knowledge $\mathcal{B}$, positive target examples $\mathcal{E}^+$, and negative target examples $\mathcal{E}^-$ satifying

$$\forall e^+ \in \mathcal{E}^+,\ \ \mathcal{H} \wedge \mathcal{B} \models e^+ \qquad\qquad \forall e^- \in \mathcal{E}^-,\ \ \mathcal{H} \wedge \mathcal{B} \not\models e^- \tag{1}$$

where $\mathcal{B}, \mathcal{E}^+, \mathcal{E}^-$ are sets of facts. Intuitively, eq. 1 states that, from given background facts, we can *infer* all positive examples based on the rules and not any negative examples. The *inference* is accomplished by forward chaining, introduced in the next section.

3.2 Forward chaining

Forward chaining is a deduction tool to infer all true facts from the background knowledge and the given rules. Following the definition in Evans & Grefenstette (2018), for any set of ground atoms $\mathcal{A}$ and set of Horn rules $\mathcal{R}$, the facts deduced via one-step forward chaining is the following set

$$\mathcal{F}_{\mathcal{R},1}(\mathcal{A}) := \ \mathcal{A} \cup \left\{ a \ \middle|\ a \leftarrow a_1 \wedge \cdots \wedge a_k \in \mathrm{g}(\mathcal{R}), \bigwedge_{l=1}^{k} a_l \in \mathcal{A} \right\} \tag{2}$$

where $\mathrm{g}(\mathcal{R})$ denotes all the *ground rules* obtained by substituting constants for all the variables in a rule of $\mathcal{R}$. Then, we recursively define the facts deduced through $t \geq 2$ steps $\mathcal{F}_{\mathcal{R},t}(\mathcal{A})$

$$\mathcal{F}_{\mathcal{R},t}(\mathcal{A}) := \mathcal{F}_{\mathcal{R},1}\big(\mathcal{F}_{\mathcal{R},t-1}(\mathcal{A})\big), \text{ where } \mathcal{F}_{\mathcal{R},0}(\mathcal{A}) := \mathcal{A} \tag{3}$$

We say the *fixed point* is reached if, for some T, no new fact can be derived, i.e., $\mathcal{F}_{\mathcal{R},T}(\mathcal{A}) = \mathcal{F}_{\mathcal{R},T+1}(\mathcal{A})$. We define operator $\mathrm{ForwardChaining}(\mathcal{B}, \mathcal{R})$ that takes background knowledge and a given set of Horn rules as input and produces $\mathcal{F}_{\mathcal{R},T}(\mathcal{B})$ as output.

4 Horn Rule Network

A Horn Rule Network (HRN) is established based on a set $\mathcal{U}$ of universal meta-rules, which is simply an *implication* $\mathrm{HEAD} \leftarrow \mathrm{BODY}$. In contrast to previous works, these meta-rules are not specific templates, agnostic to any specific language bias, and expert-free. HEAD is to be unified (i.e., matching) with an unground atom and BODY is to be unified with zero, one, or more unground atoms. The predicate in each atom may have any number of arguments, ensuring the modeling flexibility. Besides, each universal meta-rule involves at most V logical variables, i.e., the unified atoms can take these variables as arguments.[2] After training, we can extract interpretable rules from the universal meta-rules by substituting the unified head atom for HEAD, substituting the conjunction of the unified body atoms for BODY, and substituting the unified variables for the arguments.

Overall, the primary goal of HRN is to learn the unification with respect to the universal meta-rules, involving unifying the unground atoms with HEAD, BODY and unifying the arguments of the unified atoms with the logical variables of the universal meta-rules. Similar to existing neuro-symbolic ILP methods, we attempt to leverage powerful continuous optimization to conduct the unification learning. **See Algorithm 1 for an overview, and Appendix B (due to page limit) for a running example.**

We first define notations and concepts (Sec. 4.1), and then describe the continuous relaxations (Sec. 4.2), continuous optimization (Sec. 4.3), extraction of interpretable rules (Sec. 4.4), and hyperparameter analysis (Sec. 4.5).

4.1 Notations and concepts

We use lowercase letters to denote predicates, e.g., $p, even$, and uppercase letters to denote hyperparameters (e.g., V). For positive integer N, we use $[N]$ to denote the set $\{1, \ldots, N\}$. We use the calligraphic font to denote sets, e.g., $\mathcal{P}, \mathcal{A}$. RV is the abbreviation for *random variable*. $[\ldots, \ldots, \ldots]$ denotes concatenation of vectors, and $\cdot$ denotes inner products.

[2] Slightly abusing the notion, we call the arguments of an atom's predicate the arguments of the atom. Similarly, we call the arity of an atom's predicate the arity of the atom.

Algorithm 1 Learning procedure

Input: background knowledge $\mathcal{B}$, target examples $\mathcal{E}^+, \mathcal{E}^-$, symbolic hyperparameters ($|\mathcal{U}|$, B, V, $|\mathcal{P}^{\text{aux}}|$, ...), neural hyperparameters
Output: interpretable logical rules
Initialize model parameters $\boldsymbol{\theta}$, set of head-candidate atoms $\mathcal{A}^{\text{h}}$ (per CWA or OWA mode), set of body-candidate atoms $\mathcal{A}^{\text{b}}$
repeat
  for all universal meta-rules **do**
    Probabilistic head atom unification between head candidates and HEAD /* Sec. 4.2.2 */
    Probabilistic body atom unification between body candidates and BODY /* Sec. 4.2.3 */
    Probabilistic variable unification between arguments and variables /* Sec. 4.2.4 */
  end for
  Probabilistic joint unification /* Sec. 4.2.5 */
  Linearly sample atoms and variables /* Sec. 4.3.2 and Algo. 2 */
  Compute gradient $\widetilde{\nabla}_{\boldsymbol{\theta}}$ and optimize $\boldsymbol{\theta}$ /* Sec. 4.3.1 & Eq. 7 */
until balanced accuracy achieves 100% or maximum iterations reached
Extract interpretable rules from the final model /* Sec. 4.4 and Algo. 3 */

Let $\mathcal{P}^{\text{bk}}$, $\mathcal{P}^{\text{aux}}$, and $\mathcal{P}^{\text{tgt}}$ denote, respectively, the sets of background, auxiliary, and target predicates. Predicate invention in HRN is accomplished simply by defining the number of auxiliary predicates $|\mathcal{P}^{\text{aux}}|$ and their arity. The semantics of these invented auxiliary predicates are determined by the learned rules (i.e., *intensional predicate*). Let $\mathcal{P} := \mathcal{P}^{\text{bk}} \cup \mathcal{P}^{\text{aux}} \cup \mathcal{P}^{\text{tgt}}$ denote the union of all predicates. Let the function $\text{arity}(p)$ denote the arity of predicate $p \in \mathcal{P}$, and let A to denote the maximum arity $\max_{p\in\mathcal{P}}(\text{arity}(p))$. Since HRN allows a predicate to occur multiple times in the body of an instantiated rule, we use M, referred to as multiplicity, to denote the maximum number of times a predicate is allowed to appear in any rule.

To learn the unifications, we define the set of *head-candidate atoms* $\mathcal{A}^{\text{h}}(\mathcal{P}) := \{p(\text{arg}_1^p, \ldots, \text{arg}_{\text{arity}(p)}^p) \,|\, p \in \mathcal{P}\}$ for each universal meta-rule.[3] Each arg refers to an argument of the corresponding predicate, and is to be unified with the universal meta-rule's logical variables. The candidate atoms in $\mathcal{A}^{\text{h}}(\mathcal{P})$ are to be unified with the meta-rule's HEAD. Similarly, we define the set of *body-candidate atoms* $\mathcal{A}^{\text{b}}(\mathcal{P}, M) := \{p(\text{arg}_1^{p,m}, \ldots, \text{arg}_{\text{arity}(p)}^{p,m}) \,|\, p \in \mathcal{P}, m \in [M]\}$ for each universal meta-rule. The definition reflects that each occurrence of the same predicate may take different arguments, ensuring high flexibility of modeling again. The candidate atoms in $\mathcal{A}^{\text{b}}(\mathcal{P})$ are to be unified with the meta-rule's BODY. The dependence on $\mathcal{P}, M$ will be omitted in the following notations as it is fixed for each ILP task.

4.2 Continuous relaxation

To learn the unifications via continuous optimization, the symbolic units are relaxed into continuous spaces. We represent the universal meta-rules, including HEAD & BODY & logical variables, and candidate atoms, including predicates & arguments, by continuous embeddings (Sec. 4.2.1). Furthermore, to leverage gradient-based optimizers, HRN's gradient should be tractable. Considering that it is difficult to make the forward chaining (Eq. 2, 3) differentiable with respect to atom unification and variable unification[4], we attempt to unify the symbolic units in a probabilistic way (Sec. 4.2.2 - 4.2.5), which introduces stochastic structures into HRN. Although stochastic structures are not directly differentiable, we can estimate unbiased gradients (Sec. 4.3.1) via sampling (Sec. 4.3.2).

4.2.1 Continuous representation

For each universal meta-rule, we associate (learnable) embeddings $\boldsymbol{\nu}^{\text{h}} \in \mathbb{R}^{d_{\text{a}}}$ and $\boldsymbol{\nu}^{\text{b}} \in \mathbb{R}^{d_{\text{a}}}$ with its HEAD and BODY, respectively. Embeddings $\boldsymbol{\nu}_1, \ldots, \boldsymbol{\nu}_V \in \mathbb{R}^{d_{\text{v}}}$ are assigned to the meta-rule's variables, reflecting that variables are unrelated across rules.

[3] Since the unifications are established meta-rule by meta-rule, we omit the meta-rule index $r \in [|\mathcal{U}|]$ for all candidate sets, embeddings, random variables defined in Section 4 to simplify notations. We will explicitly mention the meta-rule index when needed.

[4] Existing works only consider predicate unification, which can be differentiable using restricted templates.

For each head-candidate atom and body-candidate atom, we design *predicate embeddings*, *argument embeddings*, and *atom embeddings* as follows. For predicate $p \in \mathcal{P}$, we associate an embedding $\boldsymbol{\eta}_p \in \mathbb{R}^{d_\mathrm{p}}$, reflecting that predicates are shared across rules. For each head-candidate atom $a \in \mathcal{A}^\mathrm{h}$, we associate an embedding $\boldsymbol{\eta}^\mathrm{h}_{a,i} \in \mathbb{R}^{d_\mathrm{v}}$ for its i-th argument, where $i \in [\mathrm{arity}(a)]$. Similarly, for each body-candidate atom $a \in \mathcal{A}^\mathrm{b}$, we give an embedding $\boldsymbol{\eta}^\mathrm{b}_{a,i} \in \mathbb{R}^{d_\mathrm{v}}$ for its i-th argument. The argument embedding design reveals that the same predicate may take different arguments, depending on occurring in head or occurring in body and depending on the occurrence in body.

The atom embeddings are constructed from the predicates and arguments embeddings above via concatenation and (learnable) linear transformations. Each head-candidate atom $a \in \mathcal{A}^\mathrm{h}$ is assigned the embedding $\boldsymbol{\eta}^\mathrm{h}_a := \boldsymbol{W}^\mathrm{h} \cdot [\boldsymbol{\eta}_p, \boldsymbol{\eta}^\mathrm{h}_{a,1}, \ldots, \boldsymbol{\eta}^\mathrm{h}_{a,A}] \in \mathbb{R}^{d_\mathrm{a}}$, where, for $i > \mathrm{arity}(p)$, the argument embedding $\boldsymbol{\eta}^\mathrm{h}_{a,i}$ is defined as the all-zero vector of length d_v, and $\boldsymbol{W}^h$ is a $d_\mathrm{a} \times (d_\mathrm{p} + A \cdot d_\mathrm{v})$ matrix. Similarly, we define the embedding of a body-candidate atom $a \in \mathcal{A}^\mathrm{b}$ as $\boldsymbol{\eta}^\mathrm{b}_a := \boldsymbol{W}^\mathrm{b} \cdot [\boldsymbol{\eta}_p, \boldsymbol{\eta}^\mathrm{b}_{a,1}, \ldots, \boldsymbol{\eta}^\mathrm{b}_{a,A}] \in \mathbb{R}^{d_\mathrm{a}}$. We remark that $\boldsymbol{W}^\mathrm{h}, \boldsymbol{W}^\mathrm{b}$ are also independent across meta-rules.

Finally, we use $\boldsymbol{\theta}$ to collectively denote all the above model parameters. These various embeddings grant HRN complete ability to learn any set of Horn rules, further demonstrated in the following.

4.2.2 Probabilistic head atom unification

Recall that all the following unifications are meta-rule independent, in line with the property that head atoms are unrelated across rules, which ensures complete flexibility of our universal meta-rules. Specifically, to unify the head-candidate atoms with the meta-rule's HEAD, we define an RV a^h over the set $\mathcal{A}^\mathrm{h}$, with the categorical distribution $P_{\boldsymbol{\theta}}(\mathsf{a}^\mathrm{h} = a) = \mathrm{softmax}\left(\boldsymbol{\eta}^\mathrm{h}_a \cdot \boldsymbol{\nu}^\mathrm{h} : a \in \mathcal{A}^\mathrm{h}\right)_a$, for $a \in \mathcal{A}^\mathrm{h}$, i.e., probabilistically unifying the head atoms.[5] Such a unification reflects the property that the Horn rule only allows one head atom.

Different from previous works (Campero et al., 2018; Glanois et al., 2022), where only predicates are unified via learning embeddings, we learn to unify the entire atoms that are constituted by predicates and arguments, the same for the following body atom unifications.

In addition, since $\mathcal{A}^\mathrm{h}$ contains background predicates, new background facts can be inferred via forward chaining, which follows the OWA.[6] We refer to this setting as the OWA mode of HRN. In the CWA setting, since all correct background facts are completely given, we restrict $\mathcal{A}^\mathrm{h}$ to involve only auxiliary and target predicates $\mathcal{P}^\mathrm{aux} \cup \mathcal{P}^\mathrm{tgt}$. We refer to this setting as HRN's CWA mode.

4.2.3 Probabilistic body atom unification

To unify the body-candidate atoms with the BODY of each universal meta-rule, let $\mathbf{a}^\mathrm{b} := (\mathsf{a}^\mathrm{b}_a : a \in \mathcal{A}^\mathrm{b})$ be a vector of independent Bernoulli RVs with (σ denotes the sigmoid function)

$$P_{\boldsymbol{\theta}}(\mathsf{a}^\mathrm{b}_a = 1) := \sigma\left(\boldsymbol{\eta}^\mathrm{b}_a \cdot \boldsymbol{\nu}^\mathrm{b}\right), \forall a \in \mathcal{A}^\mathrm{b}, \tag{4}$$

The RVs indicate whether the relevant body-candidate atom is unified to the BODY.

To reduce the computational cost, we limit the number of atoms that can be unified to each BODY to be at most B, as in symbolic ILP systems (e.g., Cropper & Morel (2021a)). Let $\mathsf{s} := \sum_{a=1}^{|\mathcal{A}^\mathrm{b}|} \mathsf{a}^\mathrm{b}_a$. Then, by the independence assumption and the Bayes' rule, we have

$$P_{\boldsymbol{\theta}}\left(\mathbf{a}^\mathrm{b} | \mathsf{s} \leq B\right) = \frac{\prod_{a=1}^{|\mathcal{A}^\mathrm{b}|} P_{\boldsymbol{\theta}}(\mathsf{a}^\mathrm{b}_a) \cdot \mathbf{1}_{\{\mathsf{s} \leq B\}}}{P_{\boldsymbol{\theta}}(\mathsf{s} \leq B)}, \tag{5}$$

where $\mathbf{1}_{\{\cdot\}}$ is the indicator function. This unification reflects that body atoms are permutation-equivalent in a Horn rule (since conjunction is).

The differentiability, with respect to $\boldsymbol{\theta}$, of the left-hand side of Eq. 5 requires a differentiable computing algorithm for the denominator of the right-hand side, which is the cumulative distribution

[5] We slightly abuse a to denote both an element of $\mathcal{A}^\mathrm{h}$ (or $\mathcal{A}^\mathrm{b}$) and its index.

[6] *Closed-world assumption* (CWA) and *open-world assumption* (OWA) (Reiter, 1981) are two different setups. Most works assume CWA, where the background knowledge is complete, i.e., $\mathcal{B}$ exactly contains all *true* facts with background predicates. Our work also considers OWA, a more realistic setting, where $\mathcal{B}$ may be incomplete.

function (CDF) of a Poisson binomial RV. Ahmed et al. (2023) propose an algorithm to compute the probability mass function (PMF) of Poisson binomial RV in a differentiable way. The CDF can then be computed by summing over the PMFs.

4.2.4 PROBABILISTIC VARIABLE UNIFICATION

After unifying the atoms, we now unify the meta-rules' variables with the arguments of the unified atoms. Again, recall that the variable unifications are meta-rule independent. To this end, let $\mathbf{v}^{\mathrm{h}} := (\mathsf{v}^{\mathrm{h}}_{a,i})_{a\in\mathcal{A}^{\mathrm{h}},i\in[\mathrm{arity}(a)]}$ be a vector of independent RVs where each RV $\mathsf{v}^{\mathrm{h}}_{a,i}$ takes its values from the set $\{1,\dots,V\}$ according to the categorical distribution $P_{\boldsymbol{\theta}}(\mathsf{v}^{\mathrm{h}}_{a,i} = v) = \mathrm{softmax}\left(\boldsymbol{\eta}^{\mathrm{h}}_{a,i}\cdot\boldsymbol{\nu}_1,\dots,\boldsymbol{\eta}^{\mathrm{h}}_{a,i}\cdot\boldsymbol{\nu}_V\right)_v$, for $v\in[V]$. The RVs $\mathsf{v}^{\mathrm{h}}_{a,i}$ determine probabilistic unification between the v-th logical variable and the i-th argument of the head-candidate atom a with probability $P_{\boldsymbol{\theta}}(\mathsf{v}^{\mathrm{h}}_{a,i} = v)$.

Similarly, for body atoms, we repeat the same procedure with $\mathbf{v}^{\mathrm{b}} := (\mathsf{v}^{\mathrm{b}}_{a,i})_{a\in\mathcal{A}^{\mathrm{b}},i\in[|\mathrm{arity}(a)|]}$, where each RV $\mathsf{v}^{\mathrm{b}}_{a,i}$ is distributed according to $P_{\boldsymbol{\theta}}(\mathsf{v}^{\mathrm{b}}_{a,i} = v) = \mathrm{softmax}\left(\boldsymbol{\eta}^{\mathrm{b}}_{a,i}\cdot\boldsymbol{\nu}_1,\dots,\boldsymbol{\eta}^{\mathrm{b}}_{a,i}\cdot\boldsymbol{\nu}_V\right)_v$.

This probabilistic variable unification reflects the following properties of arguments and variables: (a) arguments in one rule share the same set of variables, whether in the head or in the body, which interact with each other in forward chaining; (b) the same predicate may have different arguments in one rule and across rule; (c) arguments are independently unified to variables, ensuring complete flexibility of our universal meta-rules again. Besides, different from previous works (Campero et al., 2018; Glanois et al., 2022), where arguments are specified manually or by expert-provided restricted proto-rules, we directly learn to unify the variables with the arguments, improving the automatic scalability to higher arity.

4.2.5 PROBABILISTIC JOINT UNIFICATION

After defining the unifications for each universal meta-rules, now we can define the joint unification. Specifically, the joint atom unification with respect to $\mathrm{HEAD}, \mathrm{BODY}$ over all meta-rules is assumed independent and is done according to the distribution $P_{\boldsymbol{\theta}}(\mathbf{a}) := \prod_{r\in[|\mathcal{U}|]} P_{\boldsymbol{\theta}}(\mathsf{a}^{\mathrm{h}}_r)\cdot P_{\boldsymbol{\theta}}(\mathbf{a}^{\mathrm{b}}_r \mid \sum_{a=1}^{|\mathcal{A}^{\mathrm{b}}|}\mathsf{a}^{\mathrm{b}}_{r,a}\le B)$. Assuming independence again, the joint variable unification is done according to $P_{\boldsymbol{\theta}}(\mathbf{v}) := \prod_{r\in[|\mathcal{U}|]} P_{\boldsymbol{\theta}}(\mathbf{v}^{\mathrm{h}}_r)\cdot P_{\boldsymbol{\theta}}(\mathbf{v}^{\mathrm{b}}_r)$. These RVs with r index are the same as aforementioned except that we add back the meta-rule index r.

4.3 CONTINUOUS OPTIMIZATION

To search for semantically correct hypotheses, we maximize the following objective

$$L(\boldsymbol{\theta}) := \mathbb{E}_{\mathbf{a}\sim P_{\boldsymbol{\theta}}(\mathbf{a})}\left[\mathbb{E}_{\mathbf{v}\sim P_{\boldsymbol{\theta}}(\mathbf{v})}\left[f(\mathbf{a},\mathbf{v})\right]\right] \tag{6}$$

to learn the unifications. Specifically, after sampling $\mathbf{a},\mathbf{v}$ from $P_{\boldsymbol{\theta}}(\mathbf{a}), P_{\boldsymbol{\theta}}(\mathbf{v})$, each universal meta-rule is instantiated to a specific Horn rule. Leveraging these rules, we can predict the truth values of target facts via forward chaining $\mathrm{ForwardChaining}(\mathcal{B},\mathcal{R}(\mathbf{a},\mathbf{v}))$, where $\mathcal{R}(\mathbf{a},\mathbf{v})$ denotes the rule set instantiated according to $\mathbf{a},\mathbf{v}$. Since the truth values are usually imbalanced for the target predicate, we use balanced accuracy (Brodersen et al., 2010) between *true* and *false* to evaluate the correctness of the instantiated rules, computed by $(TP/P + TN/N)/2$, where P, N are the number of *true*s and *false*s corresponding to all target ground atoms[7], and TP, TN are the number of correctly predicted *true*s and *false*s. Given a specific ILP task, the balanced accuracy is thus a non-differentiable function of $\mathbf{a},\mathbf{v}$, which we denote by $f(\mathbf{a},\mathbf{v})$.

After optimizing, the distributions are expected to collapse to or approximate deterministic distributions, which deterministically instantiate universal meta-rules to specific rules that can entail the targets. Maximizing such an objective via gradient-based methods, where learnable parameters are in the sampling distribution and not in the function f, is feasible with the REINFORCE gradient estimation (Williams, 1992). Considering our proposed multi-stage unification, we design a nested variant, including a nested gradient estimator (Sec. 4.3.1) and nested sampling strategy (Sec. 4.3.2).

[7]Refer to the atoms constructed by combining the target predicate with all combinations of constants.

Algorithm 2 Linear sampling of body atoms

1: **Input:** Bernoulli parameter $P_{\boldsymbol{\theta}}(\mathsf{a}^{\mathrm{b}}_a = 1)$ for $a \in \mathcal{A}^{\mathrm{b}}$, body-candidate set $\mathcal{A}^{\mathrm{b}}$, maximum number of body atoms B
2: **Output:** $\mathrm{BODY} \sim P_{\boldsymbol{\theta}}\left(\mathbf{a}^{\mathrm{b}} | \mathsf{s} \leq B\right)$
3: $\mathrm{BODY} = [], j = B$
4: **for** $a = |\mathcal{A}^{\mathrm{b}}|$ **to** 1 **do**
5: $\quad \mathsf{a}^{\mathrm{b}}_a \sim \mathrm{Bernoulli}\left(\frac{P_{\boldsymbol{\theta}}(\mathsf{a}^{\mathrm{b}}_a=1)\cdot P_{\boldsymbol{\theta}}\left(\sum_{k=1}^{a-1}\mathsf{a}^{\mathrm{b}}_k \leq j-1\right)}{P_{\boldsymbol{\theta}}\left(\sum_{k=1}^{a}\mathsf{a}^{\mathrm{b}}_k \leq j\right)}\right)$
6: $\quad$ **if** $\mathsf{a}^{\mathrm{b}}_a = 1$ **then** $j = j - 1,\ \mathrm{BODY.append}(a)$
7: **end for**

4.3.1 GRADIENT ESTIMATOR

Leveraging the REINFORCE technique (Williams, 1992) two times sequentially, we can rewrite the gradient $\nabla \mathbb{E}_{P_{\boldsymbol{\theta}}(\mathbf{a})}[\mathbb{E}_{P_{\boldsymbol{\theta}}(\mathbf{v})}[f(\mathbf{a}, \mathbf{v})]]$ as $\mathbb{E}_{P_{\boldsymbol{\theta}}(\mathbf{a})}[\mathbb{E}_{P_{\boldsymbol{\theta}}(\mathbf{v})}[f(\mathbf{a}, \mathbf{v}) \nabla \log P_{\boldsymbol{\theta}}(\mathbf{a}, \mathbf{v})]]$. Then, using nested Monte Carlo estimator (Rainforth et al., 2018), we derive the following unbiased nested estimator

$$\widetilde{\nabla}_{\boldsymbol{\theta}} = \frac{1}{N^a} \sum_{i=1}^{N^a} \frac{1}{N^v} \sum_{j=1}^{N^v} \left(f\left(\mathbf{a}^{(i)}, \mathbf{v}^{(i,j)}(\mathbf{a}^{(i)})\right)\right) \nabla_{\boldsymbol{\theta}} \log P_{\boldsymbol{\theta}}\left(\mathbf{a}^{(i)}, \mathbf{v}^{(i,j)}(\mathbf{a}^{(i)})\right), \tag{7}$$

where $\mathbf{a}^{(i)} \sim P_{\boldsymbol{\theta}}(\mathbf{a})$ and, independently of the atom samples, $\mathbf{v}^{(i,j)} \sim P_{\boldsymbol{\theta}}(\mathbf{v})$. The dependence of the retained coordinates $\mathbf{v}^{(i,j)}(\mathbf{a}^{(i)})$ on $\mathbf{a}^{(i)}$ corresponds to the fact that, after sampling head and body atoms, only the variables belonging to these atoms contribute to forward chaining. Therefore, the irrelevant variables are removed from the log probability, i.e., $\mathbf{v}^{(i,j)}(\mathbf{a}^{(i)})$ is a subset of $\mathbf{v}^{(i,j)}$ where only the relevant variables are retained.

This estimator is still unbiased, even after removing the irrelevant variables.

Proposition 4.1. *The gradient estimator $\widetilde{\nabla}_{\boldsymbol{\theta}}$ is unbiased. (proved in Appendix A.1)*

4.3.2 SAMPLING STRATEGY

Like other REINFORCE estimators, $\widetilde{\nabla}_{\boldsymbol{\theta}}$ requires sampling from a distribution. Specifically, we need to first sample atoms $\mathbf{a} = ((\mathsf{a}^{\mathrm{h}}_r)_{r\in[|\mathcal{U}|]}, (\mathsf{a}^{\mathrm{b}}_r)_{r\in[|\mathcal{U}|]})$ and then sample $\mathbf{v} = ((\mathbf{v}^{\mathrm{h}}_r)_{r\in[|\mathcal{U}|]}, (\mathbf{v}^{\mathrm{b}}_r)_{r\in[|\mathcal{U}|]})$. Considering the independence of these RVs, sampling $\mathsf{a}^{\mathrm{h}}_r, \mathbf{v}^{\mathrm{h}}_r, \mathbf{v}^{\mathrm{b}}_r$ is straightforward. The difficulty arises from sampling $\mathbf{a}^{\mathrm{b}}_r$ by Eq. 5, where different Bernoulli RVs are sampled conditional on an inequality constraint. A naive rejection sampling works for a moderate-to-high probability of the inequality constraint; however, the time usage will be unacceptable if the probability is too small (i.e., very small acceptance probability). Existing works can handle equality constraints in linear time (Ahmed et al., 2023). We propose a similar sequential sampling process, under the inequality constraint, also achieving linear time complexity. See Algorithm 2 and its proof in Appendix A.2.

Proposition 4.2. *Algorithm 2 samples body atoms from Eq. 5 in time $\mathcal{O}(n)$.*

4.4 EXTRACTING INTERPRETABLE RULES

To extract interpretable rules from HRN, each argument takes the variable with the maximum probability, and each HEAD takes the head-candidate atom with the maximum probability. For each BODY, we also need to select the body-candidate atoms with the maximum probability of Eq. 5, which is not trivial. We propose Algorithm 3 to extract body atoms. Refer to Appendix A.3 for details. The following theorem demonstrates the sufficient capacity of HRN, proved in Appendix A.4.

Theorem 4.3 (HRN's Completeness)**.** *The HRN family can model any finite set of Horn rules.*

4.5 HYPERPARAMETERS ANALYSIS

The hyperparameters of HRN are divided into two parts, which we call *symbolic hyperparameters* and *neural hyperparameters*. Symbolic hyperparameters include the number of universal meta-rules

Table 1: Results on standard benchmarks

Task		∂ILP	LRI	HRI	DFORL	HRN
Arithmetic	Even-Odd	100	100	40	-	100
	Even-Succ2	48.5	100	40	-	100
	Fizz	10	10	0	-	100
	Buzz	35	70	40	-	100
Lists	Length	92.5	100	0	0	100
Family Tree	Grandparent	96.5	100	100	100	100
	Uncle	70	-	-	100	100
Graph	Adjacent to Red	50.5	100	100	100	100
	Two Children	95	0	100	100	100
	Graph Coloring	94.5	0	100	100	100

$|\mathcal{U}|$, maximum arity of auxiliary predicates, number of auxiliary predicates, maximum number of body atoms B, maximum number of variables V. These hyperparameters are analogous to several symbolic ILP methods (Cropper & Morel, 2021a). Neural hyperparameters include optimizer hyperparameters, entropy regularization coefficient λ, embedding dimension $d_{\mathrm{p}}, d_{\mathrm{v}}, d_{\mathrm{a}}$, sample size of the gradient estimator N^a, N^v. These are common hyperparameters in Deep Learning. Overall, the hyperparameters do not involve expert knowledge, such as hand-crafted rule templates, which is in line with the proposal from Chen et al. (2025).

5 Experiments

We evaluate HRN and compare it against existing methods on both standard benchmarks with relatively simple rules and recently proposed large-scale datasets. We also conduct ablation studies, extensive hyperparameter analysis, runtime comparison (Appendix D.4), and case study (Appendix D.5). See Appendix C for implementation details.

5.1 Experiments on standard benchmarks

First, we experiment on standard ILP datasets proposed by Evans & Grefenstette (2018). The dataset contains 20 ILP tasks, involving arithmetic, list relation, family tree, and graphs. These tasks can be perfectly solved in the CWA setting with unary & binary predicates and with rules possessing less than 2 body atoms. Therefore, this small-scale datasets cannot fully evaluate HRN's performance, but can compare HRN with existing methods. We set CWA mode for HRN in these tasks.

Table 1 show the results on the tasks not fully solved by existing works.[8] Following previous works, the reported metric is the success rate over at least 10 random seeds. A trial is considered successful if, on validation set, the mean squared error between the truth value (of the target facts) and the predicted truth value is less than 1e-4. See Appendix D.1 for full results. HRN can solve all of the 20 tasks, in particular, the hardest Fizz and Buzz tasks that require learning complicated recursive rules. We analyze and compare the learned rules in Appendix D.5.

5.2 Experiments on large-scale benchmarks

Second, we test on a recently proposed large-scale (in terms of hypothesis space) noisy dataset GeoILP, which is inapplicable for many existing methods due to their restricted language bias or insufficient handling of high-arity predicates (Chen et al., 2025). We use applicable neuro-symbolic ILP models, αILP (Shindo et al., 2023) and NEUMANN (Shindo et al., 2024), and the large language model (DeepSeek & Qwen) to compare against HRN.

[8]We do not report DFORL's results on recursive tasks since it experiments in a setting where the model inputs contain all the positive target examples, which is different from baselines' setting.

Table 2: Results on large-scale benchmarks

MODEL	CIRCLE_1	CIRCLE_2	CONG_1	PARA_1	PARA_2	PARA_3
QWEN3.5	50.0 ± 0.0	50.0 ± 0.0	49.8 ± 0.2	54.9 ± 15.8	49.9 ± 0.0	49.9 ± 0.0
DEEPSEEK	80.0 ± 25.8	75.0 ± 26.4	76.9 ± 16.9	93.3 ± 16.1	93.3 ± 16.1	86.2 ± 20.8
α ILP	62.5 ± 0.0	62.5 ± 0.0	58.3 ± 0.0	50.0 ± 0.0	50.0 ± 0.0	62.5 ± 0.0
NEUMANN	$\mathbf{80.0 \pm 1.0}$	73.7 ± 2.0	54.6 ± 0.1	68.3 ± 1.4	53.3 ± 1.7	50.0 ± 0.0
HRN	$\mathbf{80.0 \pm 0.0}$	$\mathbf{80.1 \pm 2.7}$	$\mathbf{81.3 \pm 2.8}$	$\mathbf{96.5 \pm 2.9}$	$\mathbf{94.4 \pm 2.5}$	$\mathbf{98.5 \pm 1.2}$

PERP_1	EQANGLE_1	EQANGLE_2	EQRATIO_1	EQRATIO_2	EQRATIO_3	AVG.
49.9 ± 0.0	50.0 ± 0.0	50.4 ± 1.3	50.0 ± 0.0	50.0 ± 0.0	50.0 ± 0.0	50.4
80.0 ± 13.9	62.2 ± 7.0	54.7 ± 4.0	64.4 ± 23.3	72.5 ± 17.2	79.6 ± 18.0	76.5
50.0 ± 0.0	OOM	OOM	OOM	OOM	OOM	56.5
51.6 ± 0.1	OOM	OOM	OOM	OOM	OOM	61.6
$\mathbf{83.4 \pm 0.0}$	$\mathbf{72.5 \pm 6.2}$	$\mathbf{67.1 \pm 4.4}$	$\mathbf{68.0 \pm 8.8}$	$\mathbf{83.0 \pm 4.2}$	$\mathbf{90.4 \pm 5.4}$	**82.9**

We focus on the basic level of GeoILP (also infeasible by existing methods), involving predicates of up to 8 arguments and the OWA setting. Due to the high arity, the hypothesis space is far larger than the small-scale tasks, due to the combinatorial explosion in terms of arity (Cropper & Morel, 2021a). Table 2 shows the results on 12 basic-level tasks. We report balanced accuracy (see Sec. 4.3) on validation set. All results are averaged over 10 random seeds. HRN outperforms all baselines on all 12 tasks, surpassing them by at least 6.4 on average. αILP and NEUMANN suffer from out-of-memory issues for octonary predicates with 256GB memory limit. In addition, we found LLM suffers from extremely high variance on all tasks, revealing that LLM is not a robust ILP solver.

5.3 ABLATION STUDY

We conduct ablation studies to reveal the effect of CWA/OWA modes and of removing irrelevant variables in Eq. 7. Results are listed in Table 4. When studying one of the effects, we keep all other setups the same. We found that CWA & OWA modes don't have a significant performance gap. The CWA mode surpasses the OWA mode by 0.5 on average. We argue that the larger hypothesis space in OWA mode increases the optimization difficulty. Improving the optimization efficiency under OWA mode is a promising future direction. See Appendix D.2 for the CWA/OWA comparison on more tasks. Besides, we empirically prove that HRN can benefit from removing irrelevant variables, which improves balanced accuracy by 0.9 on average. In addition, removing irrelevant variables is nearly never worse than retaining them.

5.4 HYPERPARAMETER SENSIVITY ANALYSIS

Figure 1 (in Appendix D) illustrates how balanced accuracy varies with variable sample size N^v and atom sample size N^a. For a fair comparison, we keep the total number of training samples the same (Figure 1.a & Figure 1.c), i.e., increasing the sample size will decrease the number of training steps accordingly. We found that a large sample size can cause high variance and typically decrease the average balanced accuracy. To explore the reason, we keep the total number of training steps the same (Figure 1.b & Figure 1.d). In this case, a large sample size can improve the balanced accuracy and the variances typically do not increase obviously, even decrease on some tasks. The two phenomena above justifies that the superior performance of large sample sizes requires sufficient training steps. See Appendix D.3 for comprehensive sensitivity analysis on other hyperparameters.

6 CONCLUSION

We propose the Horn Rule Network (HRN), a new neuro-symbolic ILP model family that can model any set of Horn rules without expert defined restricted language biases. Notably, HRN directly learn logical variables via gradient-based optimization. Experiments on standard benchmarks with relatively simple rules show that HRN achieves the best performance. Besides, we also test HRN on large-scale ILP tasks with complex rules and found that HRN consistently outperforms baselines and LLMs. We also conduct extensive experiments to analyze HRN.

AI USE STATEMENT

In this work, we used generative AI tools for assisting in the writing of proofs and assisting with translation. We have not used generative AI tools for helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, implementing methods, supporting qualitative and thematic data analysis, and interpreting results. Generating synthetic data sets and cleaning and reformating dataset are not applicable to this work. Additionally, we used generative AI tools for summarizing or analysing existing literature, sourcing/searching for information, and identifying relevant literature. We have reviewed all AI-assisted work. We checked the correctness of proofs improved by LLMs. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Code is submitted as supplementary material to the review system. Mathematical proofs can be found in Section A.

REFERENCES

Kareem Ahmed, Zhe Zeng, Mathias Niepert, and Guy Van den Broeck. SIMPLE: A gradient estimator for k-subset sampling. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=GPJVuyX4p_h.

Aws Albarghouthi, Paraschos Koutris, Mayur Naik, and Calvin Smith. Constraint-based synthesis of datalog programs. In J. Christopher Beck (ed.), *Principles and Practice of Constraint Programming - 23rd International Conference, CP 2017, Melbourne, VIC, Australia, August 28 - September 1, 2017, Proceedings*, volume 10416 of *Lecture Notes in Computer Science*, pp. 689–706. Springer, 2017. doi: 10.1007/978-3-319-66158-2_44. URL https://doi.org/10.1007/978-3-319-66158-2_44.

Laura Antanas, Plinio Moreno, and Luc De Raedt. Relational kernel-based grasping with numerical features. In Katsumi Inoue, Hayato Ohwada, and Akihiro Yamamoto (eds.), *Inductive Logic Programming - 25th International Conference, ILP 2015, Kyoto, Japan, August 20-22, 2015, Revised Selected Papers*, volume 9575 of *Lecture Notes in Computer Science*, pp. 1–14. Springer, 2015. doi: 10.1007/978-3-319-40566-7_1. URL https://doi.org/10.1007/978-3-319-40566-7_1.

Sándor Bartha and James Cheney. Towards meta-interpretive learning of programming language semantics. In Dimitar Kazakov and Can Erten (eds.), *Inductive Logic Programming - 29th International Conference, ILP 2019, Plovdiv, Bulgaria, September 3-5, 2019, Proceedings*, volume 11770 of *Lecture Notes in Computer Science*, pp. 16–25. Springer, 2019. doi: 10.1007/978-3-030-49210-6_2. URL https://doi.org/10.1007/978-3-030-49210-6_2.

Yoshua Bengio, Ian Goodfellow, Aaron Courville, et al. *Deep learning*, volume 1. MIT press Cambridge, MA, USA, 2017.

David A Bohan, Corinne Vacher, Alireza Tamaddoni-Nezhad, Alan Raybould, Alex J Dumbrell, and Guy Woodward. Next-generation global biomonitoring: large-scale, automated reconstruction of ecological networks. *Trends in ecology & evolution*, 32(7):477–487, 2017.

Kay Henning Brodersen, Cheng Soon Ong, Klaas Enno Stephan, and Joachim M. Buhmann. The balanced accuracy and its posterior distribution. In *2010 20th International Conference on Pattern Recognition*, pp. 3121–3124, 2010. doi: 10.1109/ICPR.2010.764.

Andres Campero, Aldo Pareja, Tim Klinger, Josh Tenenbaum, and Sebastian Riedel. Logical rule induction and theory learning using neural theorem proving. *CoRR*, abs/1809.02193, 2018. URL http://arxiv.org/abs/1809.02193.

Zahra Chaghazardi, Saber Fallah, and Alireza Tamaddoni-Nezhad. Trustworthy vision for autonomous vehicles: A robust logic-infused deep learning approach. In *2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2869–2874, 2024. doi: 10.1109/ITSC58415.2024.10919652.

Si Chen, Richong Zhang, and Xu Zhang. Geoilp: A synthetic dataset to guide large-scale rule induction. In *The Thirteenth International Conference on Learning Representations*, 2025.

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, 2023. URL https://openreview.net/forum?id=F8VKQyDgRVj.

Andrew Cropper. Learning logic programs though divide, constrain, and conquer. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pp. 6446–6453. AAAI Press, 2022. doi: 10.1609/AAAI.V36I6.20596. URL https://doi.org/10.1609/aaai.v36i6.20596.

Andrew Cropper and Sebastijan Dumancic. Inductive logic programming at 30: A new introduction. *J. Artif. Intell. Res.*, 74:765–850, 2022. doi: 10.1613/JAIR.1.13507. URL https://doi.org/10.1613/jair.1.13507.

Andrew Cropper and Céline Hocquette. Learning logic programs by discovering where not to search. In Brian Williams, Yiling Chen, and Jennifer Neville (eds.), *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pp. 6289–6296. AAAI Press, 2023. doi: 10.1609/AAAI.V37I5.25774. URL https://doi.org/10.1609/aaai.v37i5.25774.

Andrew Cropper and Rolf Morel. Learning programs by learning from failures. *Mach. Learn.*, 110(4): 801–856, 2021a. doi: 10.1007/S10994-020-05934-Z. URL https://doi.org/10.1007/s10994-020-05934-z.

Andrew Cropper and Rolf Morel. Predicate invention by learning from failures. *arXiv preprint arXiv:2104.14426*, 2021b.

Andrew Cropper and Stephen H. Muggleton. Can predicate invention compensate for incomplete background knowledge? In Slawomir Nowaczyk (ed.), *Thirteenth Scandinavian Conference on Artificial Intelligence - SCAI 2015, Halmstad, Sweden, November 5-6, 2015*, volume 278 of *Frontiers in Artificial Intelligence and Applications*, pp. 27–36. IOS Press, 2015. doi: 10.3233/978-1-61499-589-0-27. URL https://doi.org/10.3233/978-1-61499-589-0-27.

Andrew Cropper, Sebastijan Dumancic, and Stephen H. Muggleton. Turning 30: New ideas in inductive logic programming. In Christian Bessiere (ed.), *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pp. 4833–4839. ijcai.org, 2020a. doi: 10.24963/IJCAI.2020/673. URL https://doi.org/10.24963/ijcai.2020/673.

Andrew Cropper, Richard Evans, and Mark Law. Inductive general game playing. *Mach. Learn.*, 109 (7):1393–1434, 2020b. doi: 10.1007/S10994-019-05843-W. URL https://doi.org/10.1007/s10994-019-05843-w.

Wang-Zhou Dai, Stephen H. Muggleton, Jing Wen, Alireza Tamaddoni-Nezhad, and Zhi-Hua Zhou. Logical vision: One-shot meta-interpretive learning from real images. In Nicolas Lachiche and Christel Vrain (eds.), *Inductive Logic Programming - 27th International Conference, ILP 2017, Orléans, France, September 4-6, 2017, Revised Selected Papers*, volume 10759 of *Lecture Notes in Computer Science*, pp. 46–62. Springer, 2017. doi: 10.1007/978-3-319-78090-0_4. URL https://doi.org/10.1007/978-3-319-78090-0_4.

DeepSeek-AI. Deepseek-v3 technical report, 2024. URL https://arxiv.org/abs/2412.19437.

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.

Richard Evans, José Hernández-Orallo, Johannes Welbl, Pushmeet Kohli, and Marek Sergot. Making sense of sensory input. *Artificial Intelligence*, 293:103438, 2021.

Richard Evans, Matko Bosnjak, Lars Buesing, Kevin Ellis, David Pfau, Pushmeet Kohli, and Marek J Sergot. Making sense of raw input. In *IJCAI*, 2022.

Jonathan Feldstein, Dominic Phillips, and Efthymia Tsamoura. Efficiently learning probabilistic logical models by cheaply ranking mined rules. *CoRR*, abs/2409.16238, 2024. doi: 10.48550/ARXIV.2409.16238. URL https://doi.org/10.48550/arXiv.2409.16238.

Mikhail Galkin, Priyansh Trivedi, Gaurav Maheshwari, Ricardo Usbeck, and Jens Lehmann. Message passing for hyper-relational knowledge graphs. *arXiv preprint arXiv:2009.10847*, 2020.

Kun Gao, Hanpin Wang, Yongzhi Cao, and Katsumi Inoue. Learning from interpretation transition using differentiable logic programming semantics. *Mach. Learn.*, 111(1):123–145, 2022. doi: 10.1007/S10994-021-06058-8. URL https://doi.org/10.1007/s10994-021-06058-8.

Kun Gao, Katsumi Inoue, Yongzhi Cao, and Hanpin Wang. A differentiable first-order rule learner for inductive logic programming (abstract reprint). In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI 2024, Jeju, South Korea, August 3-9, 2024*, pp. 8478. ijcai.org, 2024a. URL https://www.ijcai.org/proceedings/2024/950.

Kun Gao, Katsumi Inoue, Yongzhi Cao, and Hanpin Wang. A differentiable first-order rule learner for inductive logic programming. *Artif. Intell.*, 331:104108, 2024b. doi: 10.1016/J.ARTINT.2024.104108. URL https://doi.org/10.1016/j.artint.2024.104108.

Kun Gao, Katsumi Inoue, Yongzhi Cao, Hanpin Wang, and Yang Feng. Differentiable rule induction from raw sequence inputs. In *The Thirteenth International Conference on Learning Representations*, 2025.

Claire Glanois, Zhaohui Jiang, Xuening Feng, Paul Weng, Matthieu Zimmer, Dong Li, Wulong Liu, and Jianye Hao. Neuro-symbolic hierarchical rule induction. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 7583–7615. PMLR, 2022. URL https://proceedings.mlr.press/v162/glanois22a.html.

Saiping Guan, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. Link prediction on n-ary relational data. In *The world wide web conference*, pp. 583–593, 2019.

Céline Hocquette, Andreas Niskanen, Matti Järvisalo, and Andrew Cropper. Learning MDL logic programs from noisy data. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pp. 10553–10561. AAAI Press, 2024. doi: 10.1609/AAAI.V38I9.28925. URL https://doi.org/10.1609/aaai.v38i9.28925.

Katsumi Inoue, Andrei Doncescu, and Hidetomo Nabeshima. Completing causal networks by meta-level abduction. *Machine Learning*, 91(2):239–277, May 2013.

Rama Kaalia, Ashwin Srinivasan, Amit Kumar, and Indira Ghosh. ILP-assisted de novo drug design. *Machine Learning*, 103(3):309–341, June 2016.

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Jonathan Lajus, Luis Galárraga, and Fabian M. Suchanek. Fast and exact rule mining with AMIE 3. In Andreas Harth, Sabrina Kirrane, Axel-Cyrille Ngonga Ngomo, Heiko Paulheim, Anisa Rula, Anna Lisa Gentile, Peter Haase, and Michael Cochez (eds.), *The Semantic Web - 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings*, Lecture Notes in Computer Science, pp. 36–52. Springer, 2020. doi: 10.1007/978-3-030-49461-2_3. URL https://doi.org/10.1007/978-3-030-49461-2_3.

Mark Law, Alessandra Russo, and Krysia Broda. The ILASP system for inductive learning of answer set programs. *CoRR*, abs/2005.00904, 2020. URL https://arxiv.org/abs/2005.00904.

Swann Legras, Céline Rouveirol, and Véronique Ventos. The game of bridge: A challenge for ILP. In Fabrizio Riguzzi, Elena Bellodi, and Riccardo Zese (eds.), *Inductive Logic Programming - 28th International Conference, ILP 2018, Ferrara, Italy, September 2-4, 2018, Proceedings*, volume 11105 of *Lecture Notes in Computer Science*, pp. 72–87. Springer, 2018. doi: 10.1007/978-3-319-99960-9_5. URL https://doi.org/10.1007/978-3-319-99960-9_5.

John W Lloyd. *Foundations of logic programming*. Springer Science & Business Media, 2012.

Jaron Maene and Luc De Raedt. Soft-unification in deep probabilistic logic. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

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*, Proceedings of Machine Learning Research, pp. 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.

Stephen H. Muggleton, Luc De Raedt, David Poole, Ivan Bratko, Peter A. Flach, Katsumi Inoue, and Ashwin Srinivasan. ILP turns 20 - biography and future challenges. *Mach. Learn.*, 86 (1):3–23, 2012. doi: 10.1007/S10994-011-5259-2. URL https://doi.org/10.1007/s10994-011-5259-2.

Yifei Peng, Yaoli Liu, Enbo Xia, Yu Jin, Wang-Zhou Dai, Zhong Ren, Yao-Xiang Ding, and Kun Zhou. Abductive logical rule induction by bridging inductive logic programming and multimodal large language models. *CoRR*, abs/2509.21874, 2025. doi: 10.48550/ARXIV.2509.21874. URL https://doi.org/10.48550/arXiv.2509.21874.

Meng Qu, Jun-Kun Chen, Louis-Pascal A. C. Xhonneux, Yoshua Bengio, and Jian Tang. Rnnlogic: Learning logic rules for reasoning on knowledge graphs. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=tGZu6DlbreV.

Tom Rainforth, Robert Cornish, Hongseok Yang, and Andrew Warrington. On nesting monte carlo estimators. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4264–4273. PMLR, 2018. URL http://proceedings.mlr.press/v80/rainforth18a.html.

Raymond Reiter. On closed world data bases. In *Readings in artificial intelligence*, pp. 119–140. Elsevier, 1981.

Tim Rocktäschel and Sebastian Riedel. End-to-end differentiable proving. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 3788–3800, 2017.

Claude Sammut, Raymond Sheh, Adam Haber, and Handy Wicaksono. The robot engineer. In Katsumi Inoue, Hayato Ohwada, and Akihiro Yamamoto (eds.), *Late Breaking Papers of the 25th International Conference on Inductive Logic Programming, Kyoto University, Kyoto, Japan, August 20th to 22nd, 2015*, volume 1636 of *CEUR Workshop Proceedings*, pp. 101–106. CEUR-WS.org, 2015. URL https://ceur-ws.org/Vol-1636/paper-10.pdf.

Hikaru Shindo, Masaaki Nishino, and Akihiro Yamamoto. Differentiable inductive logic programming for structured examples. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pp. 5034–5041. AAAI Press, 2021. doi: 10.1609/AAAI.V35I6.16637. URL https://doi.org/10.1609/aaai.v35i6.16637.

Hikaru Shindo, Viktor Pfanschilling, Devendra Singh Dhami, and Kristian Kersting. αilp: thinking visual scenes as differentiable logic programs. *Mach. Learn.*, 112(5):1465–1497, 2023. doi: 10.1007/S10994-023-06320-1. URL https://doi.org/10.1007/s10994-023-06320-1.

Hikaru Shindo, Viktor Pfanschilling, Devendra Singh Dhami, and Kristian Kersting. Learning differentiable logic programs for abstract visual reasoning. *Mach. Learn.*, 113(11):8533–8584, 2024. doi: 10.1007/S10994-024-06610-2. URL https://doi.org/10.1007/s10994-024-06610-2.

Xujie Si, Mukund Raghothaman, Kihong Heo, and Mayur Naik. Synthesizing datalog programs using numerical relaxation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 6117–6124. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/847. URL https://doi.org/10.24963/ijcai.2019/847.

Aishwarya Sivaraman, Tianyi Zhang, Guy Van den Broeck, and Miryung Kim. Active inductive logic programming for code search. In Joanne M. Atlee, Tevfik Bultan, and Jon Whittle (eds.), *Proceedings of the 41st International Conference on Software Engineering, ICSE 2019, Montreal, QC, Canada, May 25-31, 2019*, pp. 292–303. IEEE / ACM, 2019. doi: 10.1109/ICSE.2019.00044. URL https://doi.org/10.1109/ICSE.2019.00044.

Ashwin Srinivasan, Tirtharaj Dash, A Baskar, Sanjay Kumar Dey, and Mainak Banerjee. Identifying a logical specification and a program for an llm-based generator of lead molecules. *bioRxiv*, 2025. doi: 10.1101/2025.02.14.634875. URL https://www.biorxiv.org/content/early/2025/02/16/2025.02.14.634875.

Dany Varghese, Didac Barroso-Bergada, David A. Bohan, and Alireza Tamaddoni-Nezhad. Efficient abductive learning of microbial interactions using meta inverse entailment. In Stephen H. Muggleton and Alireza Tamaddoni-Nezhad (eds.), *Inductive Logic Programming - 31st International Conference, ILP 2022, Windsor Great Park, UK, September 28-30, 2022, Proceedings*, volume 13779 of *Lecture Notes in Computer Science*, pp. 127–141. Springer, 2022. doi: 10.1007/978-3-031-55630-2_10. URL https://doi.org/10.1007/978-3-031-55630-2_10.

Quan Wang, Haifeng Wang, Yajuan Lyu, and Yong Zhu. Link prediction on n-ary relational facts: A graph-based approach. *arXiv preprint arXiv:2105.08476*, 2021.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256, 1992. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Jundong Xu, Hao Fei, Meng Luo, Qian Liu, Liangming Pan, William Yang Wang, Preslav Nakov, Mong-Li Lee, and Wynne Hsu. Aristotle: Mastering logical reasoning with A logic-complete decompose-search-resolve framework. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 3052–3075. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.acl-long.153/.

Fan Yang, Zhilin Yang, and William W Cohen. Differentiable learning of logical rules for knowledge base reasoning. *Advances in neural information processing systems*, 30, 2017.

Xiaocheng Yang, Bingsen Chen, and Yik-Cheung Tam. Arithmetic reasoning with LLM: Prolog generation & permutation. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pp. 699–710, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-short.61. URL https://aclanthology.org/2024.naacl-short.61/.

Zheng Zhang, Levent Onder Yilmaz, and Bo Liu. A critical review of inductive logic programming techniques for explainable AI. *IEEE Trans. Neural Networks Learn. Syst.*, 35(8):10220–10236, 2024. doi: 10.1109/TNNLS.2023.3246980. URL https://doi.org/10.1109/TNNLS.2023.3246980.

A THEORETICAL PROOFS

A.1 UNBIASEDNESS OF THE PROPOSED GRADIENT ESTIMATOR

We prove that $\mathbb{E}[\widetilde{\nabla}_{\boldsymbol{\theta}}] = \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$ for the estimator in Eq. 7, where the expectation is over all samples used by the estimator. For a fixed realization of $\mathbf{a}$, let $\mathbf{v}(\mathbf{a})$ denote the coordinates of $\mathbf{v}$ belonging to the selected atoms, as in Section 4.3.1, and let $\mathbf{v}_-$ denote the remaining coordinates. The partition into these two sets of coordinates depends on $\mathbf{a}$, so we condition on $\mathbf{a}$ when using their independence. By the independence assumptions in Sections 4.2.4 and 4.2.5,

$$\begin{aligned} P_{\boldsymbol{\theta}}(\mathbf{a}, \mathbf{v}) &= P_{\boldsymbol{\theta}}(\mathbf{a}) P_{\boldsymbol{\theta}}\big(\mathbf{v}(\mathbf{a}) \mid \mathbf{a}\big) P_{\boldsymbol{\theta}}(\mathbf{v}_- \mid \mathbf{a}) \\ &= P_{\boldsymbol{\theta}}\big(\mathbf{a}, \mathbf{v}(\mathbf{a})\big) P_{\boldsymbol{\theta}}(\mathbf{v}_- \mid \mathbf{a}). \end{aligned}$$

Here $P_{\boldsymbol{\theta}}(\mathbf{a}, \mathbf{v}(\mathbf{a}))$ is the joint probability after marginalizing out the irrelevant coordinates. Its atom factor $P_{\boldsymbol{\theta}}(\mathbf{a})$ includes the conditional body-atom distributions and their normalizing factors in Eq. 5. Differentiating at fixed realizations therefore gives

$$\nabla_{\boldsymbol{\theta}} \log P_{\boldsymbol{\theta}}\big(\mathbf{a}, \mathbf{v}(\mathbf{a})\big) = \nabla_{\boldsymbol{\theta}} \log P_{\boldsymbol{\theta}}(\mathbf{a}, \mathbf{v}) - \nabla_{\boldsymbol{\theta}} \log P_{\boldsymbol{\theta}}(\mathbf{v}_- \mid \mathbf{a}).$$

Only the conditional score of the irrelevant variables is removed; the atom score remains in the estimator. Conditional independence and normalization imply

$$\begin{aligned} &\mathbb{E}\big[\nabla_{\boldsymbol{\theta}} \log P_{\boldsymbol{\theta}}(\mathbf{v}_- \mid \mathbf{a}) \;\big|\; \mathbf{a}, \mathbf{v}(\mathbf{a})\big] \\ &\quad = \sum_{\mathbf{v}_-} P_{\boldsymbol{\theta}}(\mathbf{v}_- \mid \mathbf{a}) \nabla_{\boldsymbol{\theta}} \log P_{\boldsymbol{\theta}}(\mathbf{v}_- \mid \mathbf{a}) \\ &\quad = \nabla_{\boldsymbol{\theta}} \sum_{\mathbf{v}_-} P_{\boldsymbol{\theta}}(\mathbf{v}_- \mid \mathbf{a}) = \mathbf{0}. \end{aligned}$$

The sums range over the realizations of the irrelevant coordinates for the fixed $\mathbf{a}$. Since these coordinates do not appear in the instantiated rules, they do not affect forward chaining, and hence $f(\mathbf{a}, \mathbf{v}) = f(\mathbf{a}, \mathbf{v}(\mathbf{a}))$. Multiplying the preceding conditional expectation by this value of f and taking expectations shows that

$$\begin{aligned} &\mathbb{E}_{P_{\boldsymbol{\theta}}(\mathbf{a},\mathbf{v})}\big[f\big(\mathbf{a}, \mathbf{v}(\mathbf{a})\big) \nabla_{\boldsymbol{\theta}} \log P_{\boldsymbol{\theta}}\big(\mathbf{a}, \mathbf{v}(\mathbf{a})\big)\big] \\ &\quad = \mathbb{E}_{P_{\boldsymbol{\theta}}(\mathbf{a},\mathbf{v})}[f(\mathbf{a}, \mathbf{v}) \nabla_{\boldsymbol{\theta}} \log P_{\boldsymbol{\theta}}(\mathbf{a}, \mathbf{v})] \\ &\quad = \nabla_{\boldsymbol{\theta}} \sum_{\mathbf{a},\mathbf{v}} P_{\boldsymbol{\theta}}(\mathbf{a}, \mathbf{v}) f(\mathbf{a}, \mathbf{v}) = \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}). \end{aligned}$$

The last equality follows from Eq. 6: f has no direct dependence on $\boldsymbol{\theta}$, and the sum is over a finite support that is independent of $\boldsymbol{\theta}$ for fixed hyperparameters.

Finally, for each $i \in [N^a]$ and $j \in [N^v]$, the full pair $(\mathbf{a}^{(i)}, \mathbf{v}^{(i,j)})$ has distribution $P_{\boldsymbol{\theta}}(\mathbf{a}, \mathbf{v})$. Thus every summand in Eq. 7 has expectation $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$, and linearity of expectation gives

$$\mathbb{E}[\widetilde{\nabla}_{\boldsymbol{\theta}}] = \frac{1}{N^a} \sum_{i=1}^{N^a} \frac{1}{N^v} \sum_{j=1}^{N^v} \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}).$$

Although summands with the same i share $\mathbf{a}^{(i)}$, their independence is not required for this argument. This proves unbiasedness.

A.2 CORRECTNESS OF ALGORITHM 2

In this section, we prove that the samples obtained from Algorithm 2, which has linear time complexity, follow the left-hand side of Eq. 5. Mathematically, this problem is similar to the Proposition 2 in Ahmed et al. (2023), except that the constraint is an inequality in our case. Our proof below is analogous to the proof in Ahmed et al. (2023).

The left-hand side of Eq. 5 can be decomposed as

$$\prod_{a=1}^{|\mathcal{A}^{\mathrm{b}}|} P\left(\mathsf{a}_a^{\mathrm{b}} \;\middle|\; \mathsf{a}_{a+1}^{\mathrm{b}}, \ldots, \mathsf{a}_{|\mathcal{A}^{\mathrm{b}}|}^{\mathrm{b}}, \sum_{k=1}^{|\mathcal{A}^{\mathrm{b}}|} \mathsf{a}_k^{\mathrm{b}} \le B\right).$$

Algorithm 3 Extracting body atoms

1: **Input:** Bernoulli parameter $P_{\boldsymbol{\theta}}(\mathsf{a}^{\mathrm{b}}_a = 1)$ for $a \in \mathcal{A}^{\mathrm{b}}$, body-candidate set $\mathcal{A}^{\mathrm{b}}$, maximum number of body atoms B
2: **Output:** $\mathbf{a}^{\mathrm{b}} = \arg\max_{\mathbf{a}^{\mathrm{b}}} P_{\boldsymbol{\theta}}\left(\mathbf{a}^{\mathrm{b}} | \mathsf{s} \leq B\right)$
3: Rank $P_{\boldsymbol{\theta}}(\mathsf{a}^{\mathrm{b}}_a = 1)$ for $a \in \mathcal{A}^{\mathrm{b}}$ in decreasing order
4: Select $\mathcal{A}^{+} := \{a \mid P_{\boldsymbol{\theta}}(\mathsf{a}^{\mathrm{b}}_a = 1) > 0.5\}$
5: **while** $|\mathcal{A}^{+}| > B$ **do**
6: Remove $a := \arg\min_{a \in \mathcal{A}^{+}} P_{\boldsymbol{\theta}}(\mathsf{a}^{\mathrm{b}}_a = 1)$ from $\mathcal{A}^{+}$
7: **end while**

Next, we will further decompose this conditional sampling probability.

Assume that RVs $\mathsf{a}^{\mathrm{b}}_{a+1}, \ldots, \mathsf{a}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|}$ have been sampled according to Algorithm 2 and that their values are $\hat{\mathsf{a}}^{\mathrm{b}}_{a+1}, \ldots, \hat{\mathsf{a}}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|}$. Let $j = B - \sum_{k=a+1}^{|\mathcal{A}^{\mathrm{b}}|} \hat{\mathsf{a}}^{\mathrm{b}}_k$. Consider the following conditional sampling probability

$$
\begin{aligned}
&P\left(\mathsf{a}^{\mathrm{b}}_a = 1 \,\middle|\, \mathsf{a}^{\mathrm{b}}_{a+1} = \hat{\mathsf{a}}^{\mathrm{b}}_{a+1}, \ldots, \mathsf{a}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|} = \hat{\mathsf{a}}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|}, \sum_{k=1}^{|\mathcal{A}^{\mathrm{b}}|} \mathsf{a}^{\mathrm{b}}_k \leq B\right) \\
&= P\left(\mathsf{a}^{\mathrm{b}}_a = 1 \,\middle|\, \mathsf{a}^{\mathrm{b}}_{a+1} = \hat{\mathsf{a}}^{\mathrm{b}}_{a+1}, \ldots, \mathsf{a}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|} = \hat{\mathsf{a}}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|}, \sum_{k=1}^{a} \mathsf{a}^{\mathrm{b}}_k \leq j\right) \\
&= \frac{P\left(\mathsf{a}^{\mathrm{b}}_a = 1\,,\ \mathsf{a}^{\mathrm{b}}_{a+1} = \hat{\mathsf{a}}^{\mathrm{b}}_{a+1}, \ldots, \mathsf{a}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|} = \hat{\mathsf{a}}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|}, \sum_{k=1}^{a} \mathsf{a}^{\mathrm{b}}_k \leq j\right)}{P\left(\mathsf{a}^{\mathrm{b}}_{a+1} = \hat{\mathsf{a}}^{\mathrm{b}}_{a+1}, \ldots, \mathsf{a}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|} = \hat{\mathsf{a}}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|}, \sum_{k=1}^{a} \mathsf{a}^{\mathrm{b}}_k \leq j\right)} \\
&= \frac{P\left(\mathsf{a}^{\mathrm{b}}_a = 1\,,\ \mathsf{a}^{\mathrm{b}}_{a+1} = \hat{\mathsf{a}}^{\mathrm{b}}_{a+1}, \ldots, \mathsf{a}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|} = \hat{\mathsf{a}}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|}, \sum_{k=1}^{a-1} \mathsf{a}^{\mathrm{b}}_k \leq j-1\right)}{P\left(\mathsf{a}^{\mathrm{b}}_{a+1} = \hat{\mathsf{a}}^{\mathrm{b}}_{a+1}, \ldots, \mathsf{a}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|} = \hat{\mathsf{a}}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|}, \sum_{k=1}^{a} \mathsf{a}^{\mathrm{b}}_k \leq j\right)} \\
&= \frac{P\left(\mathsf{a}^{\mathrm{b}}_a = 1\right) \cdot P\left(\mathsf{a}^{\mathrm{b}}_{a+1} = \hat{\mathsf{a}}^{\mathrm{b}}_{a+1}, \ldots, \mathsf{a}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|} = \hat{\mathsf{a}}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|}\right) \cdot P\left(\sum_{k=1}^{a-1} \mathsf{a}^{\mathrm{b}}_k \leq j-1\right)}{P\left(\mathsf{a}^{\mathrm{b}}_{a+1} = \hat{\mathsf{a}}^{\mathrm{b}}_{a+1}, \ldots, \mathsf{a}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|} = \hat{\mathsf{a}}^{\mathrm{b}}_{|\mathcal{A}^{\mathrm{b}}|}\right) \cdot P\left(\sum_{k=1}^{a} \mathsf{a}^{\mathrm{b}}_k \leq j\right)} \\
&= \frac{P(\mathsf{a}^{\mathrm{b}}_a = 1) \cdot P\left(\sum_{k=1}^{a-1} \mathsf{a}^{\mathrm{b}}_k \leq j-1\right)}{P\left(\sum_{k=1}^{a} \mathsf{a}^{\mathrm{b}}_k \leq j\right)}
\end{aligned}
$$

The second last step benefits from RVs' independence. The final step corresponds to the Bernoulli parameter in Algorithm 2.

A.3 Correctness of algorithm 3

Algorithm 3 corresponds to the following problem: given Bernoulli RVs $\mathsf{b}^{(i)}$ with different Bernoulli parameters $p_1, \ldots, p_n$, how to find a realization that maximizes their joint probability under the constraints $\sum_{i=1}^{n} \mathsf{b}^{(i)} \leq B$. Their joint log probability is

$$
\begin{aligned}
\log \prod_{i=1}^{n} p_i^{\mathsf{b}^{(i)}} (1-p_i)^{1-\mathsf{b}^{(i)}} &= \sum_{i=1}^{n} \mathsf{b}^{(i)} \log p_i + (1-\mathsf{b}^{(i)}) \log(1-p_i) \\
&= \sum_{i=1}^{n} \mathsf{b}^{(i)} \log \frac{p_i}{1-p_i} + \log(1-p_i)
\end{aligned}
$$

Since p_i are given, the maximization problem turns into

$$
\arg\max_{\mathsf{b}^{(1)}, \ldots, \mathsf{b}^{(n)}} \sum_{i=1}^{n} \mathsf{b}^{(i)} \log \frac{p_i}{1-p_i}
$$

Obviously, if without the constraint, we should set all the $\mathsf{b}^{(i)}$ with $\log \frac{p_i}{1-p_i} > 0$ to 1. That is, select the variables with Bernoulli parameters greater than 0.5. With the constraints, since $\frac{p_i}{1-p_i}$ grows monotonically with respect to p_i, we should first sort p_is, and set $\mathsf{b}^{(i)}$ to 1 for the top B p_i satisfying $\log \frac{p_i}{1-p_i} > 0$. This results in Algorithm 3.

A.4 Logical completeness of HRN

Theorem 4.3 claims that HRN has full expressivity for function-free Horn rules.[9] Formally, given a finite set of Horn rules $\mathcal{R}$, there exists a HRN model, parameterized by $\boldsymbol{\theta}$, from which we can extract $\mathcal{R}$ according to Section 4.4.

As preparation, we first set the hyperparameters and relevant concepts. Let the number of universal meta-rules equal to the number of Horn rules, i.e., $|\mathcal{U}| = |\mathcal{R}|$. Let V equal to the maximum number of variables for any rule in $\mathcal{R}$, B equal to the maximum number of body atoms for any rule in $\mathcal{R}$. Let $\mathcal{P}$ be all the predicates appearing in $\mathcal{R}$. Let M, the multiplicity of body atoms, be the maximum number of times a predicate appears in the body of any single rule in $\mathcal{R}$. Let A be the maximum arity for all the predicates in $\mathcal{P}$. In addition, let J denote the maximum number of arguments for any rule in $\mathcal{R}$. As in Sec. 4.1, let $\mathcal{A}^{\mathrm{h}}(\mathcal{P}) := \{p(\arg_1^p, \ldots, \arg_{\mathrm{arity}(p)}^p) \,|\, p \in \mathcal{P}\}$ be the set of head-candidate atoms, $\mathcal{A}^{\mathrm{b}}(\mathcal{P}, M) := \{p(\arg_1^{p,m}, \ldots, \arg_{\mathrm{arity}(p)}^{p,m}) \,|\, p \in \mathcal{P}, m \in [M]\}$ be the set of body-candidate atoms.[10]

In the following, we are going to construct $\boldsymbol{\theta}$ based on $\mathcal{R}$. Intuitively, since each given rule has deterministic atoms and variables, we simply set $\boldsymbol{\theta}$ to let all the unification probability values corresponding to this rule's atoms and variables be the maximums. Observing that the model parameters are meta-rule-independent, except for predicates, our constructive proof below avoids the effects of predicate embeddings so that we can capture each given rule by one meta-rule. Hence, in the following, we consider a given rule in $\mathcal{R}$, and use one meta-rule to model it.

The following proof starts by unifying variables with arguments and finish by unifying atoms.

Head variable unification Consider a given Horn rule in $\mathcal{R}$, let p denote the head atom's predicate, i denote the head atom's i-th argument, and v^* denote the index of the logical variable this argument takes.[11] Besides, let j denote the number index of that argument among all the atoms' arguments in the given rule (including body atoms' arguments).[12] By definition, there exists exactly one element in $\mathcal{A}^{\mathrm{h}}(\mathcal{P})$ related to p. Let a denote this element. Our goal is to ensure

$$v^* = \arg\max_{v \in [V]} P_{\boldsymbol{\theta}}(\mathsf{v}_{a,i}^{\mathrm{h}} = v)$$

by setting appropriate embeddings. As in the main text, we omit the meta-rule index here. Recall that

$$P_{\boldsymbol{\theta}}(\mathsf{v}_{a,i}^{\mathrm{h}} = v) = \mathrm{softmax}\left(\boldsymbol{\eta}_{a,i}^{\mathrm{h}} \cdot \boldsymbol{\nu}_1, \ldots, \boldsymbol{\eta}_{a,i}^{\mathrm{h}} \cdot \boldsymbol{\nu}_V\right)_v.$$

Therefore, it is sufficient to let $\boldsymbol{\eta}_{a,i}^{\mathrm{h}} \cdot \boldsymbol{\nu}_{v^*} = 1$ and $\boldsymbol{\eta}_{a,i}^{\mathrm{h}} \cdot \boldsymbol{\nu}_v = 0$ for $v \neq v^*$. To this end, we set $d_{\mathrm{v}} = J + 1$ (dimension of variable embeddings and argument embeddings), set variable embeddings

$$\begin{aligned} \boldsymbol{\nu}_{v^*,j} &= 1 \\ \boldsymbol{\nu}_{v,j} &= 0, \quad \forall v \neq v^* \\ \boldsymbol{\nu}_{v,J+1} &= 0, \quad \forall v \in [V], \end{aligned}$$

and set the argument embeddings

$$\begin{aligned} \boldsymbol{\eta}_{a,i,j}^{\mathrm{h}} &= 1 \\ \boldsymbol{\eta}_{a,i,j'}^{\mathrm{h}} &= 0, \quad \forall j' \in [J], j' \neq j. \end{aligned}$$

[9] Without loss of forward chaining results, functions can be replaced with predicates.

[10] If $\mathcal{R}$ is a solution of an ILP task, we may set $\mathcal{P}^{\mathrm{bk}}, \mathcal{P}^{\mathrm{aux}}, \mathcal{P}^{\mathrm{tgt}}$ separately and have $\mathcal{P} := \mathcal{P}^{\mathrm{bk}} \cup \mathcal{P}^{\mathrm{aux}} \cup \mathcal{P}^{\mathrm{tgt}}$, as in Sec. 4.1. The following proof holds for any given $\mathcal{P}$. In CWA mode, the head-candidate atoms are limited to $\mathcal{P}^{\mathrm{aux}} \cup \mathcal{P}^{\mathrm{tgt}}$, and the following proof still holds.

[11] The variable indexes can be arbitrarily set.

[12] The argument indexes can be arbitrarily set.

The new index appended to the subscript of each embedding indicates the elements. The element index starts from 1. The above setup has determined the first J elements of the argument embedding $\boldsymbol{\eta}^{\mathrm{h}}_{a,i}$ and the j-th and $(J+1)$-th element of all the variable embeddings $\boldsymbol{\nu}_1, \ldots, \boldsymbol{\nu}_V$, and has ensured $\boldsymbol{\eta}^{\mathrm{h}}_{a,i} \cdot \boldsymbol{\nu}_{v^*} = 1$ and $\boldsymbol{\eta}^{\mathrm{h}}_{a,i} \cdot \boldsymbol{\nu}_v = 0$ for $v \neq v^*$. After setting the variable embeddings for other arguments in the same way (including body atoms' arguments, see below), the variable embeddings are also fully determined.[13] The $(J+1)$-th element of the argument embeddings are reserved for atom unification and will be determined in atom unification. Consequently, setting the $(J+1)$-th element of the variable embeddings to 0 avoids the influence of the $(J+1)$-th element of the argument embeddings.

Body variable unification Setting variable embeddings and argument embeddings according to body variable unification is almost the same as head variable unification. While the head variable unification only considers one atom, the head atom, body variable unification considers every body atom in the same way. The critical difference arises from $\mathcal{A}^{\mathrm{b}}(\mathcal{P}, M)$. Different from $\mathcal{A}^{\mathrm{h}}(\mathcal{P})$, there exists multiple elements in $\mathcal{A}^{\mathrm{b}}(\mathcal{P}, M)$ related to p if $M > 1$. To deterministically select an element from $\mathcal{A}^{\mathrm{b}}(\mathcal{P}, M)$, we can index the occurrence of p in the body of the given rule.[14] Considering one of its occurrence, say, the m-th occurrence, we select $a := p(\arg_1^{p,m}, \ldots, \arg_{\mathrm{arity}(p)}^{p,m})$ from $\mathcal{A}^{\mathrm{b}}(\mathcal{P}, M)$. Then, every thing is the same as in head variable unification.

Head atom unification Again, consider a given Horn rule in $\mathcal{R}$, let p denote the head atom's predicate. By definition, there exists exactly one element in $\mathcal{A}^{\mathrm{h}}(\mathcal{P})$ related to p. Let a^* denote this element. Our goal is to ensure

$$a^* = \arg\max_{a \in \mathcal{A}^{\mathrm{h}}} P_{\boldsymbol{\theta}}(\mathsf{a}^{\mathrm{h}} = a)$$

by setting appropriate embeddings. Again, we omit the meta-rule index here. Recall that

$$P_{\boldsymbol{\theta}}(\mathsf{a}^{\mathrm{h}} = a) = \mathrm{softmax}\left(\boldsymbol{\eta}^{\mathrm{h}}_a \cdot \boldsymbol{\nu}^{\mathrm{h}} : a \in \mathcal{A}^{\mathrm{h}}(\mathcal{P})\right)_a$$

and

$$\boldsymbol{\eta}^{\mathrm{h}}_a := \boldsymbol{W}^{\mathrm{h}} \cdot [\boldsymbol{\eta}_p, \boldsymbol{\eta}^{\mathrm{h}}_{a,1}, \ldots, \boldsymbol{\eta}^{\mathrm{h}}_{a,A}] \in \mathbb{R}^{d_{\mathrm{a}}}.$$

Therefore, it is sufficient to let $\boldsymbol{\eta}^{\mathrm{h}}_{a^*} \cdot \boldsymbol{\nu}^{\mathrm{h}} = 1$ and $\boldsymbol{\eta}^{\mathrm{h}}_a \cdot \boldsymbol{\nu}^{\mathrm{h}} = 0$ for $a \neq a^*$. For simplicity, we set $d_{\mathrm{a}} = d_{\mathrm{p}} + A \cdot d_{\mathrm{v}}$ and set $\boldsymbol{W}^{\mathrm{h}}$ to be the identity matrix. We set the embedding of HEAD

$$\begin{aligned} \boldsymbol{\nu}^{\mathrm{h}}_{d_{\mathrm{p}}+J+1} &= 1 \\ \boldsymbol{\nu}^{\mathrm{h}}_{j'} &= 0, \quad \forall j' \neq d_{\mathrm{p}} + J + 1 \end{aligned}$$

and set argument embeddings

$$\begin{aligned} \boldsymbol{\eta}^{\mathrm{h}}_{a^*,1,J+1} &= 1 \\ \boldsymbol{\eta}^{\mathrm{h}}_{a,1,J+1} &= 0, \quad \forall a \neq a^* \\ \boldsymbol{\eta}^{\mathrm{h}}_{a,k,J+1} &= 0, \quad \forall a \in \mathcal{A}^{\mathrm{h}}(\mathcal{P}), k \in \{2, \ldots, A\} \end{aligned}$$

As mentioned before, the $(J+1)$-th element of argument embeddings are reserved for atom unification, and only the first argument plays the important role here. The above setup ensures $\boldsymbol{\eta}^{\mathrm{h}}_{a^*} \cdot \boldsymbol{\nu}^{\mathrm{h}} = 1$ and $\boldsymbol{\eta}^{\mathrm{h}}_a \cdot \boldsymbol{\nu}^{\mathrm{h}} = 0$ for $a \neq a^*$. As mentioned before, we avoid the effect from predicate embeddings and they can be set to any values.

Body atom unification Again, consider a given Horn rule in $\mathcal{R}$. As mentioned in body variable unification, we can select exactly one element from $\mathcal{A}^{\mathrm{b}}(\mathcal{P}, M)$ for each body atom of the given rule. Let $\mathcal{A}$ denote the set of all these elements for the rule. According to Algorithm 3, our goal is to ensure

$$\begin{aligned} P_{\boldsymbol{\theta}}(\mathsf{a}^{\mathrm{b}}_a = 1) &> 0.5, \quad \forall a \in \mathcal{A} \\ P_{\boldsymbol{\theta}}(\mathsf{a}^{\mathrm{b}}_a = 1) &\leq 0.5, \quad \forall a \notin \mathcal{A} \end{aligned}$$

by setting appropriate embeddings. Again, we omit the meta-rule index here. Recall that

$$P_{\boldsymbol{\theta}}(\mathsf{a}^{\mathrm{b}}_a = 1) := \sigma\left(\boldsymbol{\eta}^{\mathrm{b}}_a \cdot \boldsymbol{\nu}^{\mathrm{b}}\right), \forall a \in \mathcal{A}^{\mathrm{b}}(\mathcal{P}, M)$$

[13] Some rules may have fewer than J arguments. Simply set the redundant elements of variables embeddings to 0.

[14] The occurrence indexes can be arbitrarily set.

and

$$\boldsymbol{\eta}_a^{\mathrm{b}} := \boldsymbol{W}^{\mathrm{b}} \cdot [\boldsymbol{\eta}_p, \boldsymbol{\eta}_{a,1}^{\mathrm{b}}, \ldots, \boldsymbol{\eta}_{a,A}^{\mathrm{b}}] \in \mathbb{R}^{d_{\mathrm{a}}}$$

Therefore, it is sufficient to let $\boldsymbol{\eta}_a^{\mathrm{b}} \cdot \boldsymbol{\nu}^{\mathrm{b}} = 1$ for $a \in \mathcal{A}$ and $\boldsymbol{\eta}_a^{\mathrm{b}} \cdot \boldsymbol{\nu}^{\mathrm{b}} = 0$ for $a \notin \mathcal{A}$. Recall that we have set $d_{\mathrm{a}} = d_{\mathrm{p}} + A \cdot d_{\mathrm{v}}$. Similar to $\boldsymbol{W}^{\mathrm{h}}$, we set $\boldsymbol{W}^{\mathrm{b}}$ to be the identity matrix. Similar to head atom unification, we set the embedding of BODY

$$\begin{aligned} \boldsymbol{\nu}_{d_{\mathrm{p}}+J+1}^{\mathrm{b}} &= 1 \\ \boldsymbol{\nu}_{j'}^{\mathrm{b}} &= 0, \quad \forall j' \neq d_{\mathrm{p}} + J + 1 \end{aligned}$$

and set argument embeddings

$$\begin{aligned} \boldsymbol{\eta}_{a,1,J+1}^{\mathrm{b}} &= 1, \quad \forall a \in \mathcal{A} \\ \boldsymbol{\eta}_{a,1,J+1}^{\mathrm{b}} &= 0, \quad \forall a \notin \mathcal{A} \\ \boldsymbol{\eta}_{a,k,J+1}^{\mathrm{b}} &= 0, \quad \forall a \in \mathcal{A}^{\mathrm{h}}(\mathcal{P}), k \in \{2, \ldots, A\} \end{aligned}$$

B RUNNING EXAMPLE

We use a small family-relation task as a running example throughout this section. Let the background predicate be $\mathsf{parent}/2$ and the target predicate be $\mathsf{grandparent}/2$. The background knowledge is

$$\mathcal{B} = \{\mathsf{parent}(\mathsf{alice}, \mathsf{bob}), \mathsf{parent}(\mathsf{bob}, \mathsf{cara}), \mathsf{parent}(\mathsf{cara}, \mathsf{dave})\}.$$

The positive examples are

$$\mathcal{E}^+ = \{\mathsf{grandparent}(\mathsf{alice}, \mathsf{cara}), \mathsf{grandparent}(\mathsf{bob}, \mathsf{dave})\},$$

and the negative examples include, for instance,

$$\mathcal{E}^- = \{\mathsf{grandparent}(\mathsf{alice}, \mathsf{dave}), \mathsf{grandparent}(\mathsf{bob}, \mathsf{alice})\}.$$

The intended rule is

$$\mathsf{grandparent}(X, Y) \leftarrow \mathsf{parent}(X, Z) \wedge \mathsf{parent}(Z, Y).$$

We now illustrate how HRN can represent and learn this rule. Consider one universal meta-rule

$$\mathsf{HEAD} \leftarrow \mathsf{BODY},$$

with symbolic hyperparameters $B = 2$, $V = 3$, and $M = 2$. Thus, each instantiated rule may contain at most two body atoms, may use at most three logical variables, and each predicate may occur at most twice in the body. In the CWA mode, the head-candidate set contains the target atom

$$\mathcal{A}^h = \{\mathsf{grandparent}(\arg_1^g, \arg_2^g)\}.$$

The body-candidate set contains two possible occurrences of each predicate. In particular, it contains

$$\mathsf{parent}(\arg_1^{p,1}, \arg_2^{p,1}), \qquad \mathsf{parent}(\arg_1^{p,2}, \arg_2^{p,2}),$$

where the superscript denotes the occurrence index. HRN learns whether these candidate atoms should be unified with BODY and how their arguments should be unified with the variables of the universal meta-rule.

For the head atom, HRN samples

$$a^h \sim P_\theta(a^h),$$

where $P_\theta(a^h)$ is a categorical distribution over $\mathcal{A}^h$. In this example, the desired head unification assigns high probability to

$$a^h = \mathsf{grandparent}(\arg_1^g, \arg_2^g).$$

For the body atoms, HRN samples Bernoulli random variables

$$a_a^b \sim \mathrm{Bernoulli}\left(\sigma\left(\eta_a^b \cdot \nu^b\right)\right), \qquad a \in \mathcal{A}^b,$$

conditioned on selecting at most $B = 2$ atoms. The desired body selection is

$$a_{\mathsf{parent}}^b = 1, \qquad a_{\mathsf{parent}}^b = 1,$$

and all other body candidates are assigned value 0.

After atom unification, HRN samples variable unifications for the selected arguments. Let the three variables of the universal meta-rule be

$$X_1, X_2, X_3.$$

The desired variable assignment is

$$\arg_1^g \mapsto X_1, \qquad \arg_2^g \mapsto X_2,$$

for the head atom, and

$$\arg_1^{p,1} \mapsto X_1, \qquad \arg_2^{p,1} \mapsto X_3, \qquad \arg_1^{p,2} \mapsto X_3, \qquad \arg_2^{p,2} \mapsto X_2,$$

for the two selected body atoms. This sampled atom and variable unification instantiates the universal meta-rule as

$$\mathsf{grandparent}(X_1, X_2) \leftarrow \mathsf{parent}(X_1, X_3) \wedge \mathsf{parent}(X_3, X_2).$$

Renaming variables gives the intended rule

$$\mathsf{grandparent}(X, Y) \leftarrow \mathsf{parent}(X, Z) \wedge \mathsf{parent}(Z, Y).$$

Given this instantiated rule, forward chaining derives

$$\mathsf{grandparent}(\mathsf{alice}, \mathsf{cara})$$

from

$$\mathsf{parent}(\mathsf{alice}, \mathsf{bob}) \quad \text{and} \quad \mathsf{parent}(\mathsf{bob}, \mathsf{cara}),$$

and derives

$$\mathsf{grandparent}(\mathsf{bob}, \mathsf{dave})$$

from

$$\mathsf{parent}(\mathsf{bob}, \mathsf{cara}) \quad \text{and} \quad \mathsf{parent}(\mathsf{cara}, \mathsf{dave}).$$

Thus, the sampled rule correctly entails the positive examples. At the same time, it does not entail negative examples such as $\mathsf{grandparent}(\mathsf{bob}, \mathsf{alice})$.

During training, HRN repeatedly samples atom and variable unifications, evaluates the resulting rule set by forward chaining, and updates θ using the gradient estimator in Eq. 7. After optimization, the learned distributions are expected to concentrate on the desired head atom, the two desired body atoms, and the variable assignment above. The final extraction step therefore recovers the interpretable rule

$$\mathsf{grandparent}(X, Y) \leftarrow \mathsf{parent}(X, Z) \wedge \mathsf{parent}(Z, Y).$$

C Implementation details

In this section, we introduce the implementation details of HRN and LLM-based ILP.

C.1 HRN details

We use Adam optimizer Kingma (2014) to optimize the objective (Eq. 6) by computing gradient according to Eq. 7.

Hyperparameters and more implementation details, including forward chaining and evaluation, can be found in our released code.

C.2 LLM details

Specifically, we use DeepSeek-V3.1-Terminus[15] (thinking mode) (DeepSeek-AI, 2024) and Qwen3.5-Plus (thinking mode).

The prompt used for ILP tasks is given below

[15] https://huggingface.co/deepseek-ai/DeepSeek-V3.1-Terminus

```
Task description: given the following facts and positive target
examples, induce a set of Horn rules that, together with the
facts, can entail all the positive target examples and not
others. The facts and positive target examples are all Horn
facts.

Facts:
{FACTS}

Positive target examples:
{POSITIVE_TARGET_EXAMPLES}

You must format your rules as follows (variables must be capital
letters and predicates must be lowercase letters):
rmwq(X) :- coa(Y), dska(Y,Z), dska(Z,W).
ovm(Y,X) :- mbjd(Z), wquv(Z,X,Y).
...
```

`{FACTS}` is comma separated Horn facts and `{POSITIVE_TARGET_EXAMPLES}` is comma separated positive target examples.

The predicates and constants in the facts and positive target examples are converted to random strings. This follows the common practice of ILP, i.e., do not tell an ILP system the meaning of predicates and constants, which should be treated as meaningless words. Such a setting requires ILP systems to learn the semantics of the symbols (i.e., predicates, constants) completely from data.

Table 3: Full results on 20 standard ILP tasks

Task		∂ILP	LRI	HRI	DFORL	HRN
Arithmetic	Predecessor	100	100	100	100	100
	Even-Odd	100	100	40	-	100
	Even-Succ2	48.5	100	40	-	100
	LessThan	100	100	100	-	100
	Fizz	10	10	0	-	100
	Buzz	35	70	40	-	100
Lists	Length	92.5	100	0	0	100
	Member	100	100	100	100	100
Family Tree	Son	100	100	100	100	100
	Grandparent	96.5	100	100	100	100
	Husband	100	-	-	100	100
	Uncle	70	-	-	100	100
	Relatedness	100	100	100	100	100
	Father	100	100	-	100	100
Graph	Directed Edge	100	100	100	100	100
	Adjacent to Red	50.5	100	100	100	100
	Two Children	95	0	100	100	100
	Graph Coloring	94.5	0	100	100	100
	Connectedness	100	100	100	100	100
	Cyclic	100	100	100	100	100

Table 4: Comparison of CWA & OWA mode (3rd and 2nd columns) and comparison of removing & retaining irrelevant vars (3rd and 4th columns)

Task	OWA	CWA	
	remove	remove	retain
circle_1	79.3 ± 2.2	80.0 ± 0.0	80.0 ± 0.0
cong_1	81.8 ± 0.6	81.3 ± 2.8	81.6 ± 2.7
para_1	93.8 ± 6.4	96.5 ± 2.9	92.7 ± 7.0
perp_1	84.1 ± 2.2	83.4 ± 0.0	83.4 ± 0.0
Avg.	84.8	85.3	84.4

D More experimental results

D.1 Full results on 20 standard ILP tasks

Table 3 lists all results on the 20 standard ILP tasks.

D.2 Comparison between CWA and OWA

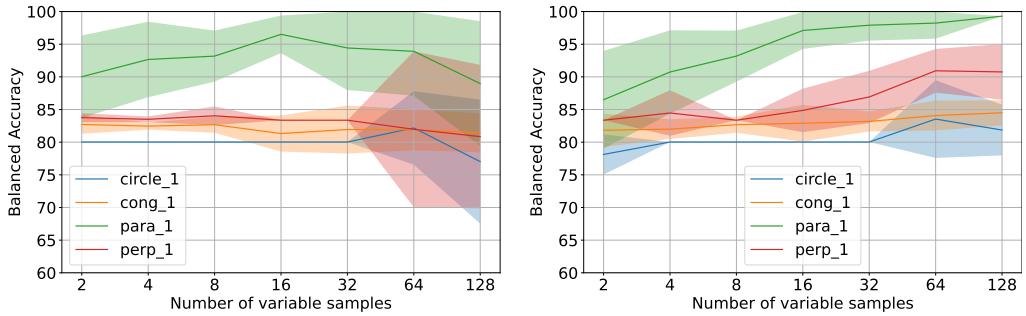
Table 5 show the comparison between HRN's CWA mode and OWA mode on more GeoILP tasks. The OWA mode shows slightly weaker performance, probably since the OWA mode suffers from a larger hypothesis space. Improving the optimization efficiency under OWA mode is a promising future direction.

D.3 Hyperparameter sensitivity analysis

Figure 2 analyzes the sensitivity of the number of universal meta-rules $|\mathcal{U}|$, maximum number of body atoms B, maximum number of variables V, and the number of auxiliary predicates $|\mathcal{P}^{\text{aux}}|$. On one hand, increasing the number of universal meta-rules and maximum number of variables can generally improve the performance. Before a threshold (say, 5 for 2.a and 4 for 2.c), the performance increases faster, since the model has insufficient capacity in this region. After the threshold, the performance gain becomes smaller, oscillating, or even negative (rare). On the other hand, increasing the maximum

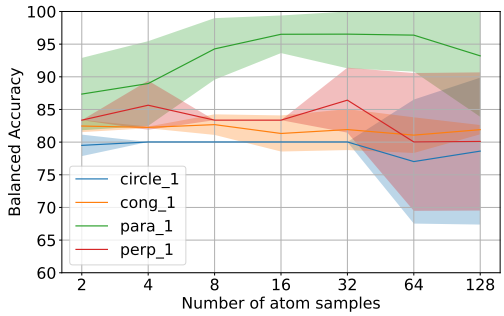
Table 5: Comparison of CWA and OWA mode on more tasks

TASK	CWA	OWA
CIRCLE_1	**80.0 ± 0.0**	79.3 ± 2.2
CIRCLE_2	**80.1 ± 2.7**	79.0 ± 2.2
CONG_1	81.3 ± 2.8	**81.8 ± 0.6**
PARA_1	**96.5 ± 2.9**	93.8 ± 6.4
PARA_2	**83.0 ± 4.2**	82.3 ± 2.0
PARA_3	**98.5 ± 1.2**	97.8 ± 1.5
PERP_1	83.4 ± 0.0	**84.1 ± 2.2**
EQANGLE_1	**72.5 ± 6.2**	70.4 ± 3.4
EQANGLE_2	67.1 ± 4.4	**69.4 ± 5.0**
EQRATIO_1	**68.0 ± 8.8**	63.3 ± 4.2
EQRATIO_2	**67.6 ± 5.5**	64.4 ± 9.2
EQRATIO_3	**66.1 ± 3.8**	63.5 ± 4.8
AVG.	**78.7**	77.4

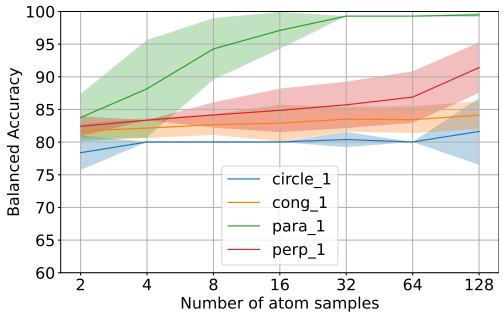


(a) Sensitivity of N^v (same #training samples)

(b) Sensitivity of N^v (same #training steps)



(c) Sensitivity of N^a (same #training samples)

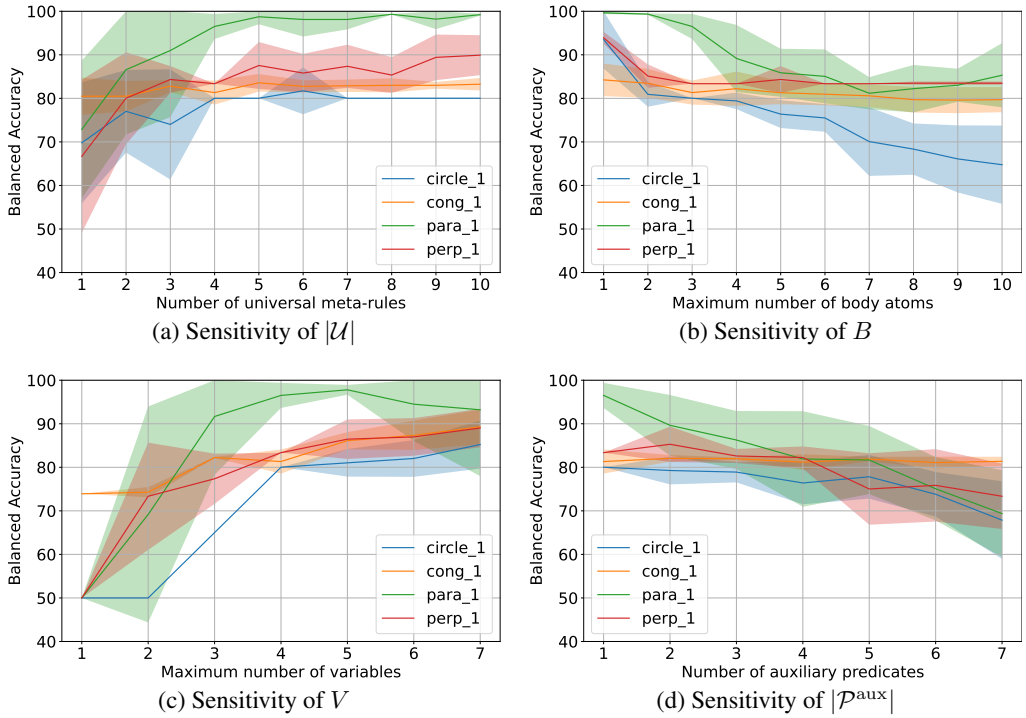


(d) Sensitivity of N^a (same #training steps)

Figure 1: Sensitivity of sample size

number of body atoms and the number of auxiliary predicates causes severe performance decrease. Considering that the hypothesis space increases when these two hyperparameters increase quickly, we argue that the performance decrease is due to insufficient optimization. The opposite performance of the two groups of hyperparameters is due to their different impacts. For example, while adding a body atom typically makes a useful Horn rule useless (i.e., not produce new facts during forward chaining since the body cannot match any existing facts), adding a new universal meta-rule may increase the possibility of successfully sampling correct rules.

Figure 3 analyzes the sensitivity of predicate embedding dimension d_{p}, variable embedding dimension d_{v}, and atom embedding dimension d_{a}. In general, the performance decreases and becomes unstable when any dimension increases. We argue that this is caused by the increased optimization difficulty. This performance drop is significantly worse for the atom embedding dimension. A small atom

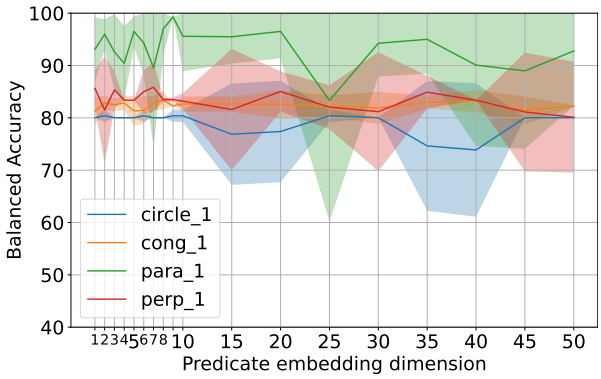


(a) Sensitivity of $|\mathcal{U}|$ (b) Sensitivity of B

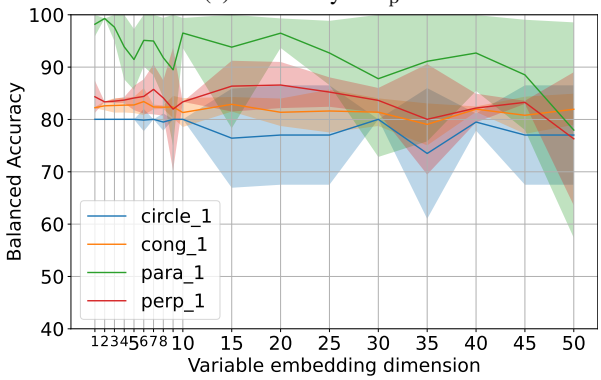
(c) Sensitivity of V (d) Sensitivity of $|\mathcal{P}^{\text{aux}}|$

Figure 2: Sensitivity of symbolic hyperparameters

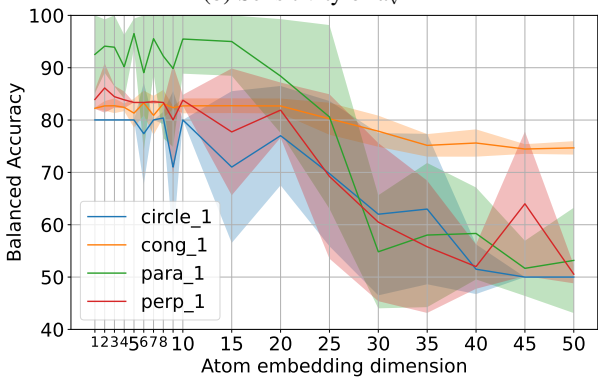
embedding dimension may effectively compress information for atoms; however, a large atom embedding dimension may increase optimization difficulty.



(a) Sensitivity of d_{p}



(b) Sensitivity of d_{v}

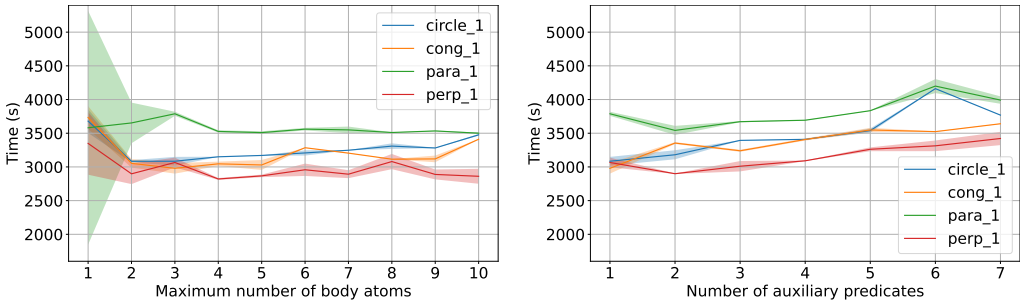


(c) Sensitivity of d_{a}

Figure 3: Sensitivity of embedding dimensions

Table 6: Time usage comparison (in seconds)

TASK	HRI	DFORL	HRN
PREDECESSOR	4 ± 0	115 ± 50	2 ± 0
LESSTHAN	20 ± 0	126 ± 106	20 ± 17
CYCLIC	69 ± 1	326 ± 163	148 ± 123
RELATEDNESS	122 ± 40	372 ± 83	63 ± 51
AVG.	54	235	58



(a) Varying B (b) Varying $|\mathcal{P}^{\text{aux}}|$

Figure 4: Runtime curves (same maximum training iterations)

D.4 TIME USAGE

Table 6 compares the time usage among HRI, DFORL, HRN on 1 CPU & 1 GPU and on comparable tasks.[16] The selected tasks have varying difficulties in terms of program size, defined by the total number of atoms in the solution provided in Evans & Grefenstette (2018). The comparison reveal that DFORL is the slowest, probably because DFORL requires an expensive preprocessing for constant substitution. HRN and HRI take about the same amount of time on average, justifying that, although HRN supports richer language bias, HRN's time consumption is still acceptable.

On GeoILP, we study how runtime varies with respect to the maximum number of body atoms B and the number of predicates. Since the number of background predicates and target predicates is fixed for a given ILP task, we vary the number of predicates by varying the number of auxiliary predicates $|\mathcal{P}^{\text{aux}}|$. Figure 4 shows that the runtime does not increase significantly due to these two factors, demonstrating HRN's scalability. The high variance on para_1 with small B is due to early stopping, which will be triggered if the model achieves 100% balanced accuracy on training data.

D.5 LEARNED RULES

HRN notably outperforms existing neuro-symbolic ILP methods on task Fizz and Buzz. We analyze the learned rules from HRN.

Fizz learns the concept of "multiples of three". HRN leans the following rules

$$fizz(X) \leftarrow zero(X)$$
$$fizz(W) \leftarrow fizz(X) \wedge succ(X, Y) \wedge succ(Y, Z) \wedge succ(Z, W)$$

To infer that the constant *3* is a multiple of three, our solution only require two forward chaining steps from the background knowledge $\{zero(0), succ(0, 1), succ(1, 2), succ(2, 3)\}$. In contrast, the solution provided by Evans & Grefenstette (2018) requires three steps. Besides, the program size (defined as the number of atoms) of our solution is seven, and the program size of the solution in Evans & Grefenstette (2018) is eleven.

[16] LLM's time usage is not comparable to others since the computing resource is probably significantly more expensive than the resource used by other methods.

Buzz learns the concept of "multiples of five". HRN leans the following rules

$$buzz(X) \leftarrow zero(X)$$
$$buzz(Z) \leftarrow buzz(X) \wedge succ2(X, Y) \wedge succ3(Y, Z)$$

To infer that the constant *10* is a multiple of five, our solution only require two forward chaining steps from the background knowledge $\{zero(0), succ2(0, 2), succ3(2, 5), ...\}$. The solution provided by Evans & Grefenstette (2018) also requires two steps. However, the program size of our solution is six, and the program size of the solution in Evans & Grefenstette (2018) is eight.

In summary, supporting richer expressiveness may help HRN learn more concise and inference-efficient solutions.