CDRL: CERTIFICATION-DRIVEN REINFORCEMENT LEARNING FOR NEUTRINO FLAVOR MODEL DISCOVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

Many scientific discovery problems require searching combinatorial hypothesis spaces under complex domain constraints. Reinforcement learning (RL) offers a promising approach, but existing methods rely on scalar rewards that provide limited information about why candidate solutions fail, leading agents to repeatedly explore invalid regions. We introduce Certification-Driven Reinforcement Learning (CDRL), a framework that leverages structured feedback from symbolic reasoning tools. When an applicable structural failure is identified, these tools produce certificates identifying the *actions* responsible. CDRL converts these certificates into reusable constraints that eliminate classes of invalid solutions and guide exploration toward valid regions. We evaluate CDRL on neutrino flavor model discovery in theoretical particle physics, where the hypothesis space exceeds 10^{26} possible models, and compare it with the state-of-the-art RL approach previously used for this task. Across three theory spaces, CDRL achieves up to $1.95\times$ higher valid model rates and up to $6.33\times$ higher neutrino model rates while evaluating up to $4\times$ fewer candidates. We further extract 40 interpretable rules from search trajectories using a post-hoc decision-tree framework and show that reusing them as soft constraints yields gains of up to $2\times$ in valid model rates and $3\times$ in neutrino model discovery across all three theory spaces. These results suggest that CDRL uncovers reusable structure in combinatorial search spaces and provides a general framework for scientific model discovery.

1 INTRODUCTION

Scientific discovery often requires exploring large spaces of candidate hypotheses. Researchers propose mathematical models, derive predictions, and compare them with experimental observations. This iterative process lies at the core of many scientific disciplines, including particle physics (Baretz et al., 2025; Feruglio & Romanino, 2021; Almumin et al., 2023; de Gouvêa et al., 2022), computational biology (Jumper et al., 2021), chemistry (Segler et al., 2018), and materials science (Bai & Zhang, 2025). In many such settings, the space of candidate models is combinatorial, and small changes in model structure can lead to dramatically different predictions. Systematically exploring these large theoretical spaces therefore becomes a major computational challenge, motivating the development of AI systems that assist scientific discovery and exploration (Merchant et al., 2023; Abramson et al., 2024; Mankowitz et al., 2023).

A common approach to exploring large hypothesis spaces is reinforcement learning (RL) (Sutton & Barto, 2018), which formulates discovery as a sequential decision-making problem. In these frameworks, an agent incrementally constructs candidate models and receives rewards based on whether they satisfy domain-specific criteria. RL has been applied successfully to automated theorem proving (Li et al., 2024), program synthesis (Zhang & Gu, 2025), and scientific discovery (Wang et al., 2023; Martín-Guerrero & Lamata, 2021; Gow et al., 2022). However, these methods typically rely on scalar reward signals that provide little information about why a candidate fails (Jha et al., 2025). Consequently, agents often waste substantial effort repeatedly exploring similar invalid regions of the combinatorial search space.

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.

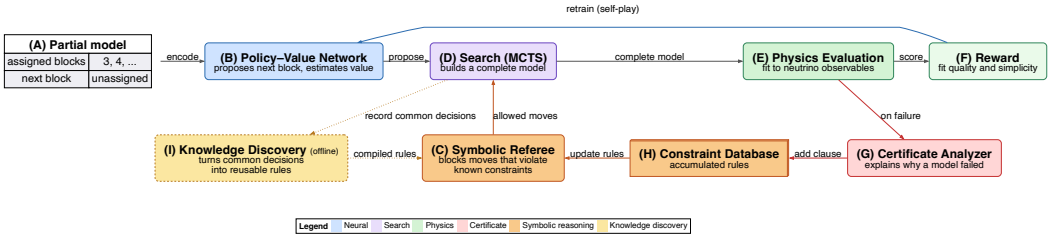


Figure 1: **The CDRL loop.** CDRL builds each model one block at a time (A) using a policy-value network and search (B, D) (Silver et al., 2016; 2017). Before the search commits to a block, a symbolic referee (C) rules out moves that are already known to violate a domain constraint, so search time is spent only on moves that remain possible. Once a model is complete, it is scored by the physics pipeline (E, F; Section 4.3 defines the score and what counts as a good model). When structural failure or symmetry redundancy is identified, the certificate analyzer (G) compiles the responsible assignments into a reusable rule stored in the constraint database (H), so the referee (C) blocks matching invalid patterns or redundant representatives after a worker imports the rule. Separately, and only after training, search knowledge discovery (I) mines high-confidence search decisions into reusable rules that bias the policy without changing symbolic legality. Section 4.2 gives the formal, SAT-based definition of the referee and constraint database, and Table 4 (Appendix B.3) works out the ways a certificate can arise. Figure 7 (Appendix F) walks through this loop step by step on a worked example.

In many scientific domains, structured feedback beyond scalar rewards is available, yet traditional RL algorithms do not fully exploit it. External reasoning tools such as theorem provers, constraint solvers, computer algebra systems, program analysis tools, and simulators can analyze candidate solutions and identify the causes of failure. Because these systems operate on formally defined objects, they can often produce *certificates* [1] that describe the outcome of their analysis and pinpoint the components responsible for violated constraints. Such certificates provide structured information that can be leveraged to guide exploration more effectively than scalar reward signals alone. Crucially, the two signals are complementary rather than competing (Figure 2, Appendix B.2).

Further, scientists have traditionally been able to distill useful heuristic rules from the problem-solving process, and these insights can accelerate future exploration of the underlying mathematical space. An RL approach that enables the extraction of such heuristic rules from the agent's decisions would therefore be valuable, both for interpretability and for improving computational efficiency in subsequent runs.

To demonstrate the potential advantages of fully incorporating such feedback, we consider models that can be constructed for neutrino masses (Baretz et al., 2025), which enables rapid exploration of mathematical models where physicists have limited intuition. Appendix A.1 provides the physics motivation and model-building context.

Certification-Driven Reinforcement Learning (CDRL) is a framework that leverages structured feedback from external reasoning tools to improve exploration in constrained combinatorial search spaces. When a structural failure can be explained, the reasoning tool returns a certificate identifying the responsible assignments. CDRL analyzes this certificate and converts the responsible assignments into a symbolic conflict clause, which is added to the global SAT constraint database (Biere et al., 2009). During search, these learned constraints are enforced by a lightweight symbolic reasoning layer that immediately eliminates partial candidates violating known domain rules, allowing the RL agent to avoid exploring impossible regions of the search space.

Appendix B.2 elaborates how certificate-based pruning complements scalar rewards and works through a concrete example.

[1] A certificate is a structured explanation produced by a reasoning tool that identifies the subset of decisions responsible for satisfying or violating domain constraints. See Section 3 for a formal definition and discussion.

Across three theory spaces for neutrino flavor model discovery, CDRL achieves up to $1.95\times$ higher valid model rates and up to $6.33\times$ higher neutrino model rates than AMBer (Baretz et al., 2025), while evaluating up to $4\times$ fewer candidates. These gains are expected to scale with the complexity of the problem and indicate substantially more efficient navigation of the underlying combinatorial hypothesis space, similar in spirit to recent advances in algorithm discovery and search-based scientific reasoning (Fawzi et al., 2022; Ruiz et al., 2025; Mankowitz et al., 2023; Abramson et al., 2024).

Beyond improved search efficiency, CDRL enables search knowledge discovery from search trajectories. Using a post-hoc decision-tree analysis of high-confidence MCTS states, we extract 40 interpretable rules that capture recurring dependencies between particle assignments. Reusing these rules as soft constraints further improves performance, suggesting that they capture meaningful and reusable structure within the theory space.

Our contributions are (i) CDRL, a framework that turns scientific evaluator feedback into shared hard constraints within policy-value-guided MCTS, integrating domain-specific certificate analysis with learned guidance and planning during model construction; (ii) state-of-the-art results on neutrino flavor model discovery, achieving substantially higher discovery rates with fewer evaluations than prior RL approaches; and (iii) a framework for extracting interpretable rules from search trajectories and reusing them as soft guidance to improve subsequent search.

2 Background and Related Work

Neutrino Flavor Model Construction. Detailed physics background is provided in Appendix A.1.

Recent work (Halverson et al., 2019; Harvey & Lukas, 2021; Nishimura et al., 2020; 2024; Carta et al., 2025; Alexander et al., 2026) has demonstrated RL as a tool for searching analogous spaces of models for various areas in particle physics. In particular, the AMBer framework (Baretz et al., 2025) integrates domain specific physics tools which allows the agent to evaluate candidate models using software used by physicists for conventional model-building. CDRL goes further, rather than using these tools only to evaluate candidates one at a time, it extracts structured certificates from them and converts these into reusable constraints that prune invalid regions of the search space, letting the system progressively accumulate knowledge about the structure of valid theories. This approach can be extended to several other applications of agent-guided workflows to scientific discovery.

Additional background on AI for scientific discovery, RL, and neuro-symbolic AI is provided in Appendix A.2.

Oracle-augmented learning. Recent work has explored integrating external reasoning tools into AI systems. Large language models have been combined with symbolic tools during inference (Kambhampati et al., 2024; Agrawal et al., 2024; Chen et al., 2020; Jana et al., 2023), while other approaches incorporate external feedback during RL (Guo et al., 2025; Jha et al., 2025). Related ideas also appear in automated reasoning and synthesis. Modern SAT solvers use conflict feedback to guide search (Liang et al., 2016), and frameworks such as CEGIS and Satisfiability Modulo Oracles refine candidates using counterexamples or constraints from external verifiers (Jha et al., 2010; Polgreen et al., 2022; Jha & Seshia, 2017). CDRL integrates domain-specific certificate analysis directly into policy-value-guided MCTS. Feedback from scientific evaluators becomes shared hard constraints enforced during model construction, coupling symbolic feedback with learned guidance and planning. Rules extracted from the resulting trajectories are then reused as soft guidance in subsequent runs.

Together, these observations motivate Certification-Driven Reinforcement Learning (CDRL), a paradigm in which symbolic reasoning systems act as structured feedback generators that guide RL over large scientific search spaces.

3 Certification-Driven Reinforcement Learning (CDRL)

Many scientific discovery tasks can be viewed as structured search problems over a large space of candidate hypotheses. An agent incrementally constructs candidate solutions while interacting with an environment that verifies whether the candidate satisfies domain constraints.

Certification-Driven Reinforcement Learning (CDRL) augments this RL loop with certificate-based constraint learning. When the verifier can explain a structural failure or symmetry redundancy, its certificate is converted into a reusable constraint on the responsible assignments.

Formally, let H denote the space of candidate hypotheses and let $\mathcal{K}_0$ denote a set of initial domain constraints known a priori. The agent maintains a growing set of constraints $\mathcal{K}$ initialized with $\mathcal{K}_0$. During training, the agent proposes candidate hypotheses using a planning procedure $\textsc{Plan}(\pi_\theta, \mathcal{K})$ (such as MCTS), which incrementally constructs candidates while enforcing constraints $\mathcal{K}$ and using π_θ to guide exploration. Each candidate is verified by an external reasoning tool. If the candidate satisfies all constraints, the agent receives a positive reward. When an applicable structural failure or symmetry redundancy is identified, a certificate is analyzed to derive a new constraint, which is added to $\mathcal{K}$ and used to prune the search space.

The overall CDRL procedure is shown in Algorithm 1 (Appendix B.1). As training progresses, the learned constraints accumulate and progressively restrict exploration to regions of the hypothesis space that remain consistent with previously discovered knowledge. This allows the RL agent to avoid rediscovering similar failure modes and focus exploration on configurations that are more likely to yield valid solutions.

Certificates and Constraint Learning. A certificate is a structured explanation produced when an applicable structural failure or symmetry redundancy is identified. It identifies the subset of decisions responsible. Let $h \in H$ denote a candidate hypothesis. When an applicable structural failure or redundancy with an already-covered equivalent hypothesis is identified for h, the verifier returns a certificate σ, which is analyzed to derive a constraint c that prevents similar configurations. This constraint is added to $\mathcal{K}$ and enforced during future exploration, progressively pruning the search space. In CDCL, conflict clauses are derived from the solver's Boolean formula. In CDRL, an external scientific pipeline evaluates complete hypotheses, and a dedicated certificate analyzer identifies representation, VEV, or symmetry assignments responsible for structural invalidity or redundancy. These assignments are translated into CNF clauses, whose soundness arguments and scope are given in Appendices B.3 and B.5. BCP enforces the clauses at every node of policy-value-guided MCTS, so certificate learning directly constrains subsequent planning and training. The ablations in Table 3 test the contributions of symbolic pruning, neural guidance, and planning. Separately, Section 5.4 extracts interpretable rules from high-confidence trajectories and reuses them as soft guidance, extending the search loop with knowledge reuse.

4 Application: Neutrino Flavor Model Discovery

We evaluate CDRL on the discovery of viable neutrino flavor models, a central problem in particle physics. The task involves constructing models that explain neutrino masses and mixing while satisfying domain constraints. We follow a common flavor symmetry model building framework, where candidate models are defined by non-Abelian symmetry choices, particle irreducible representation assignments, Abelian charges, and vacuum expectation values. This results in a large combinatorial search space, making efficient exploration critical. Figure 1 shows how the components below fit together. Further background is provided in Section 2.

4.1 Model Representation and Construction

Following the AMBer framework (Baretz et al., 2025), we encode a candidate theory as a matrix representation that captures the assignment of particles to symmetry representations and charges. Model construction is formulated as a sequential decision process. Instead of assigning every matrix entry independently, decisions are organized into blocks corresponding to logical modeling choices. The matrix encoding and block assignment order are detailed in Appendix C.

4.2 SAT-Based Constraint Layer

Rather than allowing the RL agent to freely construct candidate models, discovering only at the end that they violate domain rules, we incorporate a symbolic constraint layer that continuously enforces known scientific constraints during search. This layer rejects partial assignments that conflict with

the encoded constraints, allowing the agent to focus its exploration on the remaining regions of the combinatorial search space.

We encode the domain knowledge as a set of Boolean constraints, including exactly one representation per particle, a single active $\mathbb{Z}_N$ symmetry (a modeling choice that reduces the search space), exactly one charge per particle for that active $\mathbb{Z}_N$ symmetry, structural constraints for triplet representations, and exactly one vacuum expectation value (VEV) assignment per flavon. These constraints are compiled once into conjunctive normal form (CNF).

During the MCTS search, we use Boolean Constraint Propagation (BCP) (Davis et al., 1962; Biere et al., 2009) at every node to maintain consistency between the current partial assignment and the encoded constraints. Whenever the agent makes a new assignment, BCP detects contradictions exposed by unit propagation and infers the corresponding forced assignments. As a result, invalid branches are pruned immediately, while the state presented to the neural network is consistent under propagation of the current constraints. We therefore use BCP as a lightweight symbolic reasoning layer that filters illegal actions without requiring a full SAT solve after every decision.

4.3 RL ENVIRONMENT AND REWARD

We use the same particle physics evaluation pipeline as AMBer (Baretz et al., 2025) for direct comparison. A terminal state corresponds to a fully specified model matrix, which is passed to the physics tools. Using `Model2Mass`, the particle assignment matrix is compiled into a Lagrangian and the resulting Dirac and Majorana mass matrices are constructed, from which the light-neutrino mass matrix is obtained through the seesaw mechanism. Using `flavorpy`, the model's free parameters are then fit to neutrino observables from NuFit 5.3 (Esteban et al., 2020) and scored by a chi-squared goodness of fit χ^2. A model is *invalid* if it fails the mass-matrix rank checks (Appendix B.5) or has 50 or more free parameters. Such models receive a strong penalty. Valid models are scored by a reward that combines (i) the chi-squared fit quality and (ii) the number of free parameters n_p, normalized and clipped to a bounded range so that better fits with fewer parameters score higher (an equal weighting of the two terms by default). We designate a valid model a *neutrino model* when it additionally satisfies $\chi^2 \leq 10$ and $n_p \leq 7$. Because each evaluation requires expensive numerical optimization and dominates runtime (approx. 98%; Section 5), avoiding invalid and redundant regions of the search space before evaluation is critical. Full experimental and compute details are given in Appendix G.

4.4 CERTIFICATE EXTRACTION AND CONFLICT CLAUSES

Beyond scalar rewards, CDRL extracts structured certificates when structural invalidity or symmetry redundancy is identified and converts them into reusable *conflict clauses* added to the global constraint set $\mathcal{K}$. We implement this with a dedicated certificate-analysis routine that runs on every terminal model. It inspects the completed model matrix, tests it against the criteria below, and, when a criterion fires, identifies the exact subset of assignments responsible and emits their negation as a clause. Because a clause names the responsible assignments rather than the single candidate, one certificate prunes an entire class of models and lets the agent avoid revisiting similar configurations. We consider two main classes: (i) *insufficient-rank representation certificates*, which arise when a candidate's non-Abelian representations and VEV choice yield a rank-deficient mass matrix for every choice of Abelian charge, and eliminate that specific representation-and-VEV combination independent of the charges; and (ii) $\mathbb{Z}_N$ *symmetry certificates*, which depend only on the Abelian charges and capture two group-theoretic equivalences: charge assignments reducible to a smaller symmetry group by a shared divisor d (checked in both directions between the smaller and larger group), and charge assignments related by negation $q \mapsto -q \bmod N$, an automorphism of order 2, both of which preserve the theory's selection rule. Table 4 (Appendix B.3) demonstrates how each type of certificate works on a common $\mathbb{Z}_6$ candidate. Together, these certificates encode reusable symbolic knowledge that progressively removes invalid and redundant regions from the search space.

The construction and soundness arguments for the certificate types are given in Appendix B.3, with the scope of the failure predicate clarified in Appendix B.5.

Table 1: **Comparison with Baselines across Theory Spaces.** Valid (%) denotes structurally valid models, Neutrino (%) denotes discovered neutrino flavor models, and Min Params denotes the minimum number of free parameters found (lower is better). DNN: deep neural network (the policy-value network); PPO: Proximal Policy Optimization, the RL algorithm AMBer uses in place of CDRL's MCTS. Parenthetical multipliers under CDRL's Valid (%) and Neutrino (%) entries show improvement relative to AMBer, the prior state of the art. All other baselines (Random, ChatGPT, and the DNN/MCTS/symbolic ablations below AMBer) are shown for context and are not the basis for these multipliers. Each ablated or hybrid baseline (MCTS + symbolic feedback, DNN + symbolic feedback, Symbolic only, AlphaZero) is CDRL itself with the corresponding component removed, evaluated on the same physics pipeline described in Appendix G.3. This isolates that component's contribution rather than testing an independent reimplementation.

Method	Category	$A_4 \times \mathbb{Z}_4$			$A_4 \times \mathbb{Z}_N$			$T_{19} \times \mathbb{Z}_4$		
		Valid (%)	Neutrino (%)	Min Params	Valid (%)	Neutrino (%)	Min Params	Valid (%)	Neutrino (%)	Min Params
Uninformed and LLM baselines										
Random	Random search	0.1	0.00	-	0.1	0.00	-	0.1	0.00	-
ChatGPT (GPT-5.3) (Singh et al., 2025)	Few-shot prompting	0.2	0.00	13	0.2	0.00	18	0.1	0.00	15
Search and learning hybrids										
AlphaZero (Silver et al., 2016)	DNN + MCTS	14.9	0.01	7	16.2	0.03	7	5.4	0.04	5
MCTS + symbolic feedback	Tree search + certificates	10.2	0.00	9	9.8	0.00	10	1.1	0.00	7
DNN + symbolic feedback	Policy/value network + certificates	5.1	0.00	13	4.5	0.00	9	2.2	0.01	7
Symbolic only (Liang et al., 2016)	CDCL SAT solver	0.2	0.00	12	0.2	0.00	12	0.1	0.00	10
State-of-the-art										
AMBer (Baretz et al., 2025)	DNN + PPO	19.53	0.02	**5**	13.79	0.03	**5**	11.02	0.03	**4**
CDRL (Ours)	**DNN + MCTS + certificates**	**28.1 ± 3.3** (1.44×)	**0.10 ± 0.01** (5.00×)	**5**	**26.9 ± 3.7** (1.95×)	**0.19 ± 0.05** (6.33×)	**5**	**18.2 ± 2.1** (1.65×)	**0.09 ± 0.02** (3.00×)	**4**

4.5 Learned Clause Injection

The conflict clauses generated during evaluation are accumulated globally and injected into the SAT solver before each new search episode, requiring no recompilation since each clause is already in CNF. Detection then falls out of BCP itself. At every MCTS node the current partial assignment is fed to the solver, and to score a candidate block assignment we tentatively add its literals and propagate. Appendix B.4 details the resulting action masking and propagation.

4.6 MCTS, Neural Guidance, and Portfolio Parallelization

We search the space of partial model matrices using MCTS (Silver et al., 2016; 2017). Each node corresponds to a partially constructed candidate. At each step, a policy-value network proposes the next assignment and estimates the state's value. The policy is masked by BCP's legal-action vector (Section 4.2), ensuring only constraint-consistent actions are explored, and the masked priors guide tree traversal via PUCT (Rosin, 2011). At terminal states, the physics evaluation pipeline computes a reward that is backpropagated through the tree and used to retrain the network via self-play. This tightly couples all three components at every decision. Neural guidance proposes, symbolic reasoning prunes, and MCTS balances exploration against exploitation. Removing any one causes substantial degradation and neutrino discovery collapses to near zero without either the network or MCTS (Table 3).

To scale search, we employ portfolio-style parallelization inspired by parallel SAT solvers (Balyo et al., 2015; Menouer & Baarir, 2017) and distributed RL (Horgan et al., 2018). Multiple workers independently perform MCTS self-play while sharing (i) a replay buffer for network retraining and (ii) a global conflict clause database. Workers import batches of learned clauses and enforce them via BCP in subsequent episodes. Terminal evaluations are also cached to avoid redundant physics computation. Removing clause and replay sharing increases variance and reduces neutrino discovery (Table 3). Additional details are in Appendices G.5 and G.6.

Table 2: **Search Efficiency Comparison.** CDRL uses one-quarter of AMBer's terminal-evaluation budget. Table 1 reports CDRL means and standard deviations over four runs. Baseline counts are from Baretz et al. (2025). All counts denote inequivalent filtered models for normal ordering under AMBer's filtering and equivalence criteria. See Appendix G for count reporting and compute details.

Space	Method	Models evaluated	Neutrino models found
$A_4 \times \mathbb{Z}_4$	Random	4,000,000	28
	AMBer	4,000,000	683
	CDRL	1,000,000	**1090**
$A_4 \times \mathbb{Z}_N$	Random	4,000,000	21
	AMBer	4,000,000	1394
	CDRL	1,000,000	**2343**
$T_{19} \times \mathbb{Z}_4$	Random	24,000,000	409
	AMBer	24,000,000	6439
	CDRL	6,000,000	**7019**

4.7 Benchmark Theory Spaces

We evaluate CDRL on neutrino flavor model discovery using the same theory spaces as AMBer (Baretz et al., 2025) to ensure direct comparability. The search space consists of discrete flavor symmetry models defined by non-Abelian representations, Abelian $\mathbb{Z}_N$ charge assignments, and flavon VEV configurations. We consider three theory spaces used in prior work, defined by their symmetry structure and the number of flavons n_ϕ: (i) $A_4 \times \mathbb{Z}_4$ with $n_\phi \leq 5$ flavons, (ii) $A_4 \times \mathbb{Z}_N$ where $N \in [2, 10]$ and $n_\phi \leq 5$, and (iii) $T_{19} \times \mathbb{Z}_4$ with $n_\phi \leq 6$. These spaces increase in symmetry richness and combinatorial complexity, providing progressively harder benchmarks.

5 Experimental Results

5.1 Evaluation Metrics

We adopt the same evaluation protocol as AMBer to ensure consistency. Each candidate model is evaluated using the physics pipeline, which produces a χ^2 fit to experimental observables and the number of free parameters n_p. A model is considered valid if it satisfies structural constraints, and is considered a neutrino model if it achieves both a good fit and low complexity, defined as $\chi^2 \leq 10$ and $n_p \leq 7$. We report the percentage of valid physics models (Valid %), the percentage of viable neutrino models (Neutrino %), and the minimum number of parameters discovered.

5.2 Main Results

CDRL achieves the highest Valid (%) and Neutrino (%) of any method in every theory space, matching on AMBer's Min Params. As shown in Table 1, on $A_4 \times \mathbb{Z}_N$, CDRL achieves 26.9% valid models compared to 13.79% for AMBer (a $1.95\times$ improvement), and 0.19% neutrino models compared to 0.03% (a $6.33\times$ improvement). Similar gains are observed in other theory spaces, including up to $1.65\times$ improvements in valid model rates and up to $3\times$ improvements in neutrino discovery. These gains are achieved with substantially higher sample efficiency. As shown in Table 2, CDRL discovers more neutrino models with exactly one-quarter of AMBer's terminal-evaluation budget in each space. In combinatorial spaces exceeding 10^{26} possible models, these reductions correspond to substantially more efficient exploration. The mechanism behind these gains is concrete. Figure 1 traces how a single certificate blocks an entire class of candidates before any physics is run, and Figure 5 (Appendix E) shows an example rule recovered from the search itself (full analysis in Section 5.4).

Importance of certification-driven pruning. The additional baselines (implementation details in Appendix G.3) are CDRL itself with one component removed, and highlight the importance of combining learning, planning, and symbolic reasoning. Few-shot prompting with GPT-5.3 performs near-random search, indicating that current large language models offer little advantage in these

Table 3: **Ablation Study of CDRL Components.** Valid (%) denotes the percentage of structurally valid models, Neutrino (%) denotes the percentage of discovered neutrino flavor models, and Min Params denotes the minimum number of free parameters found.

Method / Ablation	Description	$A_4 \times \mathbb{Z}_4$			$A_4 \times \mathbb{Z}_N$			$T_{19} \times \mathbb{Z}_4$		
		Valid (%)	Neutrino (%)	Min Params	Valid (%)	Neutrino (%)	Min Params	Valid (%)	Neutrino (%)	Min Params
CDRL (Full)	DNN + MCTS + symbolic certificates + portfolio	**28.1 ± 3.3**	**0.10 ± 0.01**	**5**	**26.9 ± 3.7**	**0.19 ± 0.05**	**5**	**18.2 ± 2.1**	**0.09 ± 0.02**	**4**
No DNN	No policy/value network	10.2 ± 5.1	0.00 ± 0.00	9	9.8 ± 4.6	0.00 ± 0.00	10	1.1 ± 1.0	0.00 ± 0.00	7
No MCTS	No planning	5.1 ± 8.4	0.00 ± 0.00	13	4.5 ± 9.0	0.00 ± 0.00	9	2.2 ± 3.6	0.01 ± 0.00	7
No symbolic	No certificate	14.9 ± 3.1	0.01 ± 0.02	7	16.2 ± 2.1	0.03 ± 0.01	7	5.4 ± 4.9	0.04 ± 0.02	5
No portfolio	No clause/replay shared	26.9 ± 6.5	0.04 ± 0.02	6	25.1 ± 7.2	0.10 ± 0.09	5	12.1 ± 7.8	0.06 ± 0.04	5

underexplored theory spaces. Few-shot GPT-5.3 performs near random in our setup, suggesting that prompting alone does not substitute for guided search with symbolic feedback. Pure symbolic search, DNN + symbolic feedback, MCTS + symbolic feedback, and AlphaZero-style baselines all substantially underperform CDRL. On $A_4 \times \mathbb{Z}_N$ the AlphaZero-style baseline (MCTS and the policy-value network, without certificates) already exceeds AMBer's valid model rate (16.2% vs. 13.79%), so planning and neural guidance alone recover part of AMBer's gap to CDRL. Certification-driven pruning (Table 3) accounts for the remainder and for most of the gain in neutrino discovery. Future work may study the additional performance metrics for such rule-guided search, beyond raw number of neutrino models found.

Qualitative trends. CDRL discovers viable models spread across a substantially broader range of $\mathbb{Z}_N$ symmetry groups than AMBer, which concentrates around a few specific values of N. CDRL also contributes additional viable neutrino models beyond those reported by AMBer. Appendix D provides the full analysis, the symmetry-group and fit-quality distributions (Figures 3 and 4), and an example discovered model (Table 5).

Although CDRL introduces slight additional overhead from MCTS, certificate analysis, and symbolic propagation, the dominant computational cost (approx. 98%) arises from the physics evaluation pipeline, which is the same as AMBer. CDRL uses 32 portfolios across CPU workers and a shared GPU, whereas AMBer uses 250 parallel environments. We therefore report candidate evaluations as the primary efficiency measure. Appendix G.1 reports wall-clock and CPU usage without treating these as matched-compute results.

5.3 Ablation Study

Table 3 isolates each component's contribution by removing it from the full system. Error bars are the standard deviation across 4 runs with different random seeds. Every ablation degrades performance, confirming that the gains come from integrating learning, planning, and symbolic reasoning rather than from any single part. Removing either the neural network or MCTS collapses neutrino discovery to near zero, showing that guided exploration is necessary at all in a space this large. Removing symbolic certificates instead leaves a functioning search, with valid model rates falling from 26.9% to 16.2% and neutrino discovery from 0.19% to 0.03% on $A_4 \times \mathbb{Z}_N$, so certificates contribute a substantial improvement on top of the neural and planning components. Removing portfolio sharing lowers performance and raises variance, indicating that shared constraints and experience are critical for scaling.

5.4 Search Knowledge Discovery

A by-product of the search is a record of the decisions CDRL makes on its way to good models. We examine this record post-hoc to recover common decision patterns from the MCTS trajectories and cast them into human-readable decision trees. These decision patterns both illuminate how the agent is exploring, and with sufficient interpretation by a human physicist can provide insight on

common modes of viable models in the theory space. Our approach is related to prior work on extracting interpretable policies from RL systems (Bastani et al., 2018). However, in addition to using extracted rules purely for explanation or verification, we inject these rules as soft symbolic constraints in fresh, smaller-scale, "rule-guided" CDRL runs to test their utility in deepening exploration of fruitful regions of the theory space. Effectively, these soft constraints encourage the agent towards known viable regions of the theory space while retaining the alternative actions permitted by the hard constraints. As shown in Figure 6 (Appendix E), incorporating these rules consistently increases the number of discovered models in all three theory spaces. In $A_4 \times \mathbb{Z}_4$ and $T_{19} \times \mathbb{Z}_4$, rule-guided runs achieve $\sim 2\times$ higher valid model rates and 2–3$\times$ higher neutrino model discovery, while in $A_4 \times \mathbb{Z}_N$ we observe $\sim 1.5\times$ improvements in validity and up to $3\times$ in neutrino discovery.

The procedure used to extract soft constraints from MCTS trajectories is given in Algorithm 2 (Appendix E), with example decision trees in Figure 5. Activated rules softly reweight the policy before legality masking and MCTS, as detailed in Appendix E.

The extracted rules are simple if-else conditions over particle assignments that reveal recurring structure in successful models. Crucially, interpreting each rule within the model building process requires additional context not always captured within the decision tree, and many rules are more representative of the agent's search process than of correlations between parameters of the model. Appendix E.1 discusses these interpretation limits and examples. The full set of 40 rules is listed in Appendix H.

6 Conclusion, Limitation, and Future Work

We introduced CDRL, a framework for scientific discovery in large constrained hypothesis spaces. Unlike standard RL methods that rely primarily on scalar rewards, CDRL uses structured certificates from symbolic domain-specific reasoning tools to identify why candidates fail and convert these failures into reusable constraints. On neutrino flavor model discovery, CDRL consistently improves search efficiency across all theory spaces. Per candidate evaluated, it achieves up to $1.95\times$ higher valid model rates and $6.33\times$ higher neutrino model discovery than AMBer. Separately, reaching a comparable number of neutrino models requires up to $4\times$ fewer physics evaluations (Table 2). CDRL also achieves broader symmetry-group coverage in our experiments, matches AMBer's best model complexity while discovering far more of them, and enables post-hoc search knowledge discovery through interpretable rules extracted from search trajectories.

The core idea of CDRL is not specific to neutrino physics, nor to particle physics. Many scientific domains pair a large discrete design space with an expensive verifier and an external tool that can explain failures, including chemistry, materials science, and automated theorem proving. In each, the dominant cost is the verification step, so the same computational bottleneck we address here applies. A scalar reward wastes that expensive budget by letting the agent revisit equivalent failures, whereas a certificate removes the whole failing class once and for all. Wherever a reasoning tool can name the assignments responsible for a violation, CDRL can turn that explanation into a constraint that is reused across the search. Finally, the rules recovered through search knowledge discovery are useful beyond efficiency, as they compress what the agent learned into human-readable structure, which both accelerates future runs and gives scientists a first look at the regularities of a mathematical space they are only beginning to explore. We have not yet independently validated these rules as physically meaningful. At this stage, they describe regularities in how the search itself explores the space, which are themselves shaped by our design choices, and confirming which of them reflect genuine physical structure is left to future work with domain experts.

The main limitation of CDRL is that it depends on the availability of useful certificate-generating tools and is currently best suited to structured, discrete search spaces where constraints can be represented and reused effectively. The framework could also introduce overhead in larger search spaces from certificate analysis and clause management. Future work will extend CDRL to broader scientific domains such as chemistry and materials science, where external verifiers can provide rich feedback. Another promising direction is to learn when and how to generalize certificates into reusable constraints, and to combine CDRL with foundation models so that neural priors, symbolic pruning, and scientific verification can work together in a unified discovery loop.

AI USE STATEMENT

We used generative AI tools to assist with writing and language polishing, including improvements to grammar, phrasing, clarity, readability, and manuscript organization. This assistance was limited to presentation. The scientific ideas, methodology, implementation, experimental design, and interpretation of results were developed by the authors. The use of an LLM as an experimental baseline is described separately in Appendix G.3.1. The authors take full responsibility for the accuracy, originality, and content of the manuscript.

REPRODUCIBILITY STATEMENT

To support reproducibility, we will include the CDRL source code and the neutrino models discovered in our experiments in the supplementary materials. The model representation and construction procedure are described in Appendix C, and the CDRL algorithm and certificate mechanisms in Appendix B. Appendix G provides implementation details, network architecture, training hyperparameters, baseline descriptions, and compute resources. The KD extraction procedure and the complete rule gallery are provided in Appendices E and H, respectively.

REFERENCES

Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, et al. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, 630(8016):493–500, 2024.

Kamal Acharya, Waleed Raza, Carlos Dourado, Alvaro Velasquez, and Houbing Herbert Song. Neurosymbolic reinforcement learning and planning: A survey. *IEEE Transactions on Artificial Intelligence*, 2023.

Lakshya A Agrawal, Aditya Kanade, Navin Goyal, Shuvendu Lahiri, and Sriram Rajamani. Monitor-guided decoding of code LMs with static analysis of repository context. *Advances in Neural Information Processing Systems*, 36, 2024.

Q. R. Ahmad et al. Direct evidence for neutrino flavor transformation from neutral current interactions in the Sudbury Neutrino Observatory. *Phys. Rev. Lett.*, 89:011301, 2002. doi: 10.1103/PhysRevLett.89.011301.

M. H. Ahn et al. Indications of neutrino oscillation in a 250 km long baseline experiment. *Phys. Rev. Lett.*, 90:041801, 2003. doi: 10.1103/PhysRevLett.90.041801.

Stephon Alexander, Benjamin Bradley, Loukas Gouskos, and Cooper Niu. Autonomous Discovery of Particle Physics Theories from Experimental Data. 3 2026.

Yahya Almumin, Mu-Chun Chen, Murong Cheng, Victor Knapp-Perez, Yulun Li, Adreja Mondol, Saul Ramos-Sanchez, Michael Ratz, and Shreya Shukla. Neutrino Flavor Model Building and the Origins of Flavor and CP Violation. *Universe*, 9(12):512, 2023. doi: 10.3390/universe9120512.

Wolfgang Altmannshofer and Admir Greljo. Recent Progress in Flavor Model Building. *Ann. Rev. Nucl. Part. Sci.*, 75(1):201–322, 2025. doi: 10.1146/annurev-nucl-121423-100950.

K. S. Babu. TASI Lectures on Flavor Physics. In *Theoretical Advanced Study Institute in Elementary Particle Physics: The Dawn of the LHC Era*, pp. 49–123, 2010. doi: 10.1142/9789812838360_0002.

Xiaopeng Bai and Xingcai Zhang. Artificial intelligence-powered materials science. *Nano-micro letters*, 17(1):135, 2025.

Tomáš Balyo, Peter Sanders, and Carsten Sinz. Hordesat: A massively parallel portfolio sat solver. In *International Conference on Theory and Applications of Satisfiability Testing*, pp. 156–172. Springer, 2015.

Jason Benjamin Baretz, Max Fieg, Vijay Ganesh, Aishik Ghosh, V Knapp-Perez, Jake Rudolph, and Daniel Whiteson. Towards ai-assisted neutrino flavor theory design. *arXiv preprint arXiv:2506.08080*, 2025.

Osbert Bastani, Yewen Pu, and Armando Solar-Lezama. Verifiable reinforcement learning via policy extraction. *Advances in neural information processing systems*, 31, 2018.

Marc G Bellemare, Salvatore Candido, Pablo Samuel Castro, Jun Gong, Marlos C Machado, Subhodeep Moitra, Sameera S Ponda, and Ziyu Wang. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588(7836):77–82, 2020.

Armin Biere, Hans van Maaren, and Toby Walsh. *Handbook of satisfiability*. SAGE Publications Limited, 2009.

Federico Carta, Asa Gauntlett, Finley Griffin, and Yang-Hui He. BPS spectroscopy with reinforcement learning. , 1 2025.

Yanju Chen, Chenglong Wang, Osbert Bastani, Isil Dillig, and Yu Feng. Program synthesis using deduction-guided reinforcement learning. In *International Conference on Computer Aided Verification*, pp. 587–610. Springer, 2020.

B. T. Cleveland, Timothy Daily, Raymond Davis, Jr., James R. Distel, Kenneth Lande, C. K. Lee, Paul S. Wildenhain, and Jack Ullman. Measurement of the solar electron neutrino flux with the Homestake chlorine detector. *Astrophys. J.*, 496:505–526, 1998. doi: 10.1086/305343.

Brandon C Colelough and William Regli. Neuro-symbolic ai in 2024: A systematic review. *arXiv preprint arXiv:2501.05435*, 2025.

Martin Davis, George Logemann, and Donald Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5(7):394–397, 1962.

André de Gouvêa, Irina Mocioiu, Saori Pastore, Louis E. Strigari, L. Alvarez-Ruso, A. M. Ankowski, A. B. Balantekin, V. Brdar, M. Cadeddu, S. Carey, J. Carlson, M. C. Chen, V. Cirigliano, W. Dekens, P. B. Denton, R. Dharmapalan, L. Everett, H. Gallagher, S. Gardiner, J. Gehrlein, L. Graf, W. C. Haxton, O. Hen, H. Hergert, S. Horiuchi, P. Q. Hung, J. Isaacson, N. Jachowicz, L. Jin, A. N. Khan, A. Lovato, P. A. N. Machado, K. Mahn, D. Marfatia, C. Mariani, E. Mereghetti, J. G. Morfín, A. Nicholson, G. Paz, R. Plestid, N. Rocco, I. Sarcevic, R. Schiavilla, A. Sousa, J. Tena-Vidal, M. L. Wagman, and A. Walker-Loud. Theory of neutrino physics – snowmass tf11 (aka nf08) topical group report, 2022. URL `https://arxiv.org/abs/2209.07983`.

Jonas Degrave, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Abbas Abdolmaleki, Diego de Las Casas, et al. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2022.

Gui-Jun Ding and Davide Meloni. A Model for Tri-bimaximal Mixing from a Completely Broken A_4. *Nucl. Phys. B*, 855:21–45, 2012. doi: 10.1016/j.nuclphysb.2011.10.001.

Ivan Esteban, M. C. Gonzalez-Garcia, Michele Maltoni, Thomas Schwetz, and Albert Zhou. The fate of hints: updated global analysis of three-flavor neutrino oscillations. *JHEP*, 09:178, 2020. doi: 10.1007/JHEP09(2020)178.

Alhussein Fawzi, Matej Balog, Aja Huang, Thomas Hubert, Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Francisco J R. Ruiz, Julian Schrittwieser, Grzegorz Swirszcz, et al. Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature*, 610(7930):47–53, 2022.

Ferruccio Feruglio and Andrea Romanino. Lepton flavour symmetries, 2021. URL `https://arxiv.org/abs/1912.06028`.

Y. Fukuda et al. Evidence for oscillation of atmospheric neutrinos. *Phys. Rev. Lett.*, 81:1562–1567, 1998. doi: 10.1103/PhysRevLett.81.1562.

Murray Gell-Mann, Pierre Ramond, and Richard Slansky. Complex Spinors and Unified Theories. *Conf. Proc. C*, 790927:315–321, 1979.

Alessio Giarnetti and Davide Meloni. Reinforcement Learning Techniques for the Flavor Problem in Particle Physics. *Symmetry*, 18(1):131, 2026. doi: 10.3390/sym18010131.

S. L. Glashow. The Future of Elementary Particle Physics. *NATO Sci. Ser. B*, 61:687, 1980. doi: 10.1007/978-1-4684-7197-7_15.

Stephen Gow, Mahesan Niranjan, Samantha Kanza, and Jeremy G Frey. A review of reinforcement learning in chemistry. *Digital Discovery*, 1(5):551–567, 2022.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

James Halverson, Brent Nelson, and Fabian Ruehle. Branes with brains: exploring string vacua with deep reinforcement learning. *Journal of High Energy Physics*, 2019(6), June 2019. ISSN 1029-8479. doi: 10.1007/jhep06(2019)003. URL http://dx.doi.org/10.1007/JHEP06(2019)003.

T. R. Harvey and A. Lukas. Quark Mass Models and Reinforcement Learning. *JHEP*, 08:161, 2021. doi: 10.1007/JHEP08(2021)161.

Pascal Hitzler, Md Kamruzzaman Sarker, and Aaron Eberhart. *Compendium of Neurosymbolic Artificial Intelligence*, volume 369. IOS Press, 2023.

Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado van Hasselt, and David Silver. Distributed prioritized experience replay. corr abs/1803.00933 (2018). *arXiv preprint arXiv:1803.00933*, 2018.

Alexey Ignatiev, Antonio Morgado, and João Marques-Silva. PySAT: A Python toolkit for prototyping with SAT oracles. In *SAT*, pp. 428–437, 2018. doi: 10.1007/978-3-319-94144-8_26. URL https://doi.org/10.1007/978-3-319-94144-8_26.

Prithwish Jana, Piyush Jha, Haoyang Ju, Gautham Kishore, Aryan Mahajan, and Vijay Ganesh. Cotran: An llm-based code translator using reinforcement learning with feedback from compiler and symbolic execution. *arXiv preprint arXiv:2306.06755*, 2023.

Piyush Jha, Prithwish Jana, Pranavkrishna Suresh, Arnav Arora, and Vijay Ganesh. RLSF: Fine-tuning LLMs via symbolic feedback. *28th European Conference on Artificial Intelligence*, 2025.

Susmit Jha and Sanjit A Seshia. A theory of formal synthesis via inductive learning. *Acta Informatica*, 54(7):693–726, 2017.

Susmit Jha, Sumit Gulwani, Sanjit A Seshia, and Ashish Tiwari. Oracle-guided component-based program synthesis. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 1*, pp. 215–224, 2010.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with AlphaFold. *nature*, 596(7873):583–589, 2021.

Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Kaya Stechly, Mudit Verma, Siddhant Bhambri, Lucas Saldyt, and Anil Murthy. LLMs can't plan, but can help planning in llm-modulo frameworks. *arXiv preprint arXiv:2402.01817*, 2024.

Georgia Karagiorgi, Gregor Kasieczka, Scott Kravitz, Benjamin Nachman, and David Shih. Machine learning in the search for new fundamental physics. *Nature Reviews Physics*, 4(6):399–412, 2022.

Stephen F. King. Models of Neutrino Mass, Mixing and CP Violation. *J. Phys. G*, 42:123001, 2015. doi: 10.1088/0954-3899/42/12/123001.

Zhaoyu Li, Jialiang Sun, Logan Murphy, Qidong Su, Zenan Li, Xian Zhang, Kaiyu Yang, and Xujie Si. A survey on deep learning for theorem proving. *arXiv preprint arXiv:2404.09939*, 2024.

Jia Hui Liang, Vijay Ganesh, Pascal Poupart, and Krzysztof Czarnecki. Learning rate based branching heuristic for sat solvers. In *Theory and Applications of Satisfiability Testing–SAT 2016: 19th International Conference, Bordeaux, France, July 5-8, 2016, Proceedings 19*, pp. 123–140. Springer, 2016.

Yue Liu, Zhengwei Yang, Zhenyao Yu, Zitu Liu, Dahui Liu, Hailong Lin, Mingqing Li, Shuchang Ma, Maxim Avdeev, and Siqi Shi. Generative artificial intelligence and its applications in materials science: Current situation and future perspectives. *Journal of Materiomics*, 9(4):798–816, 2023.

Daniel J Mankowitz, Andrea Michi, Anton Zhernov, Marco Gelmi, Marco Selvi, Cosmin Paduraru, Edouard Leurent, Shariq Iqbal, Jean-Baptiste Lespiau, Alex Ahern, et al. Faster sorting algorithms discovered using deep reinforcement learning. *Nature*, 618(7964):257–263, 2023.

José D Martín-Guerrero and Lucas Lamata. Reinforcement learning and physics. *Applied Sciences*, 11(18):8589, 2021.

Alexey A Melnikov, Hendrik Poulsen Nautrup, Mario Krenn, Vedran Dunjko, Markus Tiersch, Anton Zeilinger, and Hans J Briegel. Active learning machine learns to create new quantum experiments. *Proceedings of the National Academy of Sciences*, 115(6):1221–1226, 2018.

Tarek Menouer and Souheib Baarir. Parallel learning portfolio-based solvers. *Procedia Computer Science*, 108:335–344, 2017.

Amil Merchant, Simon Batzner, Samuel S Schoenholz, Muratahan Aykol, Gowoon Cheon, and Ekin Dogus Cubuk. Scaling deep learning for materials discovery. *Nature*, 624(7990):80–85, 2023.

Lisa Messeri and Molly J Crockett. Artificial intelligence and illusions of understanding in scientific research. *Nature*, 627(8002):49–58, 2024.

Peter Minkowski. $\mu \to e\gamma$ at a Rate of One Out of 10^9 Muon Decays? *Phys. Lett. B*, 67:421–428, 1977. doi: 10.1016/0370-2693(77)90435-X.

Satsuki Nishimura, Coh Miyao, and Hajime Otsuka. Exploring the flavor structure of quarks and leptons with reinforcement learning. *JHEP*, 23:021, 2020. doi: 10.1007/JHEP12(2023)021.

Satsuki Nishimura, Coh Miyao, and Hajime Otsuka. Reinforcement learning-based statistical search strategy for an axion model from flavor. , 9 2024.

Elizabeth Polgreen, Andrew Reynolds, and Sanjit A Seshia. Satisfiability and synthesis modulo oracles. In *International Conference on Verification, Model Checking, and Abstract Interpretation*, pp. 263–284. Springer, 2022.

Christopher D Rosin. Multi-armed bandits with episode context. *Annals of Mathematics and Artificial Intelligence*, 61(3):203–230, 2011.

Francisco JR Ruiz, Tuomas Laakkonen, Johannes Bausch, Matej Balog, Mohammadamin Barekatain, Francisco JH Heras, Alexander Novikov, Nathan Fitzpatrick, Bernardino Romera-Paredes, John Van De Wetering, et al. Quantum circuit optimization with AlphaTensor. *Nature Machine Intelligence*, 7(3):374–385, 2025.

Marwin HS Segler, Mike Preuss, and Mark P Waller. Planning chemical syntheses with deep neural networks and symbolic ai. *Nature*, 555(7698):604–610, 2018.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, Oct 2017. ISSN 1476-4687. doi: 10.1038/nature24270. URL `https://doi.org/10.1038/nature24270`.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018. ISBN 0262039249.

Richard Van Noorden and Jeffrey M Perkel. Ai and science: what 1,600 researchers think. *Nature*, 621(7980):672–675, 2023.

Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, et al. Scientific discovery in the age of artificial intelligence. *Nature*, 620(7972):47–60, 2023.

George N. Wojcik, Shu Tian Eu, and Lisa L. Everett. Graph reinforcement learning for exploring model spaces beyond the standard model. *Phys. Rev. D*, 111(3):035007, 2025. doi: 10.1103/PhysRevD.111.035007.

Yongjun Xu, Xin Liu, Xin Cao, Changping Huang, Enke Liu, Sen Qian, Xingchen Liu, Yanjun Wu, Fengliang Dong, Cheng-Wei Qiu, et al. Artificial intelligence: A powerful paradigm for scientific research. *The innovation*, 2(4), 2021.

Tsutomu Yanagida. Horizontal gauge symmetry and masses of neutrinos. *Conf. Proc. C*, 7902131: 95–99, 1979.

Jingru Zhang and Bin Gu. Advances in reinforcement learning for intelligent program synthesis. In *2025 5th International Conference on Neural Networks, Information and Communication Engineering (NNICE)*, pp. 1734–1739. IEEE, 2025.

A Extended Background and Related Work

A.1 Neutrino Flavor Model Construction

There is a growing interest in using RL to construct theories in particle physics (Wojcik et al., 2025; Baretz et al., 2025; Alexander et al., 2026; Giarnetti & Meloni, 2026). A viable theory, or model, is subject to a number of consistency checks such as requiring that Lorentz invariance and other internal symmetries are respected, as well as demands that the theory reproduce certain observed qualities like the mass of a certain particle. When an RL agent attempts to construct a model, it will generically fail these checks. However, if the RL agent had feedback in the form of a certificate concerning the failure of a candidate model, it could be trained to build viable models more quickly. To demonstrate the potential advantages of fully incorporating such feedback, we consider models that can be constructed for neutrino masses (Baretz et al., 2025), which enables rapid exploration of mathematical models where physicists have limited intuition.

In our setup, an agent selects the particle content of a model and the symmetries it must satisfy. These constraints are encoded via irreducible representations of a symmetry group, yielding a Lagrangian density that defines a quantum field theory. The resulting models are evaluated using existing symbolic and numerical software for Lagrangian construction, mass-matrix extraction, and parameter fitting, followed by statistical comparison to experimental data. Exhaustive search over more than 10^{26} possible models is infeasible, as evaluating a single candidate requires non-trivial symbolic and numerical computation.

The Standard Model (SM) of particle physics describes the fundamental particles and their interactions and has exquisite agreement with experiment, with some exceptions. Perhaps the most notable exception is the prediction of massless neutrinos, but experiments (Fukuda et al., 1998; Ahmad et al., 2002; Cleveland et al., 1998; Ahn et al., 2003) have determined that they have a mass, which necessitates the presence of new particles beyond the SM. Because these new particles must couple to neutrinos, and because neutrinos and charged leptons are components of the same electroweak doublets, one generally expects them to couple to charged leptons as well. For this reason, solutions to the massive neutrino problem are typically connected to the question of flavor in the SM (Babu, 2010; Feruglio & Romanino, 2021; Altmannshofer & Greljo, 2025; King, 2015): what is the explanation for the three flavors of fermions and what is the origin for the observed levels of mixing between them?

One general framework to address these shortcomings is to add right-handed neutrinos to generate small neutrino masses via the seesaw mechanism (Minkowski, 1977; Yanagida, 1979; Glashow, 1980;

Gell-Mann et al., 1979), as well as a number of scalar particles, "flavons", whose interactions give structure to the level of lepton mixing after they acquire a certain vacuum expectation value (VEV). Adding these new particles generally results in an even larger number of free parameters than was already present in the SM, which limits the predictivity of the model. However, the parameters in the flavor sector can be strongly constrained if the model is subject to a global flavor symmetry, which gives structure to the observed mixing; in the ideal case it explains the observed values with minimal free parameters. A given theory under this framework is thus described by: the set of particles, the irreducible representations they take under the flavor symmetry, the VEV of the flavons[2], and the remaining free parameters. The number of models that exist for a given choice of flavor symmetry grows rapidly with the number of particles, and identifying solutions that fit to experiment is highly nontrivial.

A.2 Additional Related Work

AI for scientific discovery. Artificial intelligence is increasingly used across the scientific discovery pipeline, including hypothesis generation, experimental design, and large-scale data analysis (Wang et al., 2023; Xu et al., 2021). Recent advances highlight the ability of AI systems to explore large scientific search spaces, such as materials discovery using graph neural networks and generative models (Merchant et al., 2023; Liu et al., 2023), and biomolecular structure prediction using deep learning systems like AlphaFold (Abramson et al., 2024). Machine learning is also widely used in particle physics for analyzing experimental data and identifying potential new phenomena (Karagiorgi et al., 2022). Alongside these developments, there is growing interest in incorporating domain knowledge and reasoning into AI-driven discovery processes (Messeri & Crockett, 2024; Van Noorden & Perkel, 2023). In this work, we take a step in this direction by introducing CDRL, which integrates symbolic reasoning tools into the learning loop to provide structured and interpretable feedback during exploration.

Reinforcement learning for algorithm and scientific discovery. Reinforcement learning (RL) has discovered algorithms and optimized complex scientific systems, including AlphaTensor for tensor decomposition (Fawzi et al., 2022), AlphaTensor-Quantum for fault-tolerant quantum circuits (Ruiz et al., 2025), and AlphaDev for faster sorting (Mankowitz et al., 2023). It has also controlled physical systems and experiments such as tokamak plasmas (Degrave et al., 2022), stratospheric balloons (Bellemare et al., 2020), and quantum experiment design (Melnikov et al., 2018). These results show RL can navigate large combinatorial search spaces, but they depend on reward signals from a simulator or environment. CDRL instead adds symbolic reasoning tools that emit structured certificates, giving the agent richer feedback to guide exploration.

Neuro-symbolic AI. Neuro-symbolic AI combines neural learning with symbolic reasoning to strengthen reasoning, interpretability, and reliability (Hitzler et al., 2023; Acharya et al., 2023). Integration strategies range from embedding symbolic rules within neural architectures and differentiable reasoning modules to modular systems that call external reasoning tools (Colelough & Regli, 2025). CDRL takes the modular route. A learning agent queries external symbolic systems that analyze candidate solutions and return structured feedback. In the taxonomy of (Colelough & Regli, 2025), CDRL sits at the center of the neuro-symbolic design space, uniting learning, symbolic reasoning, and structured knowledge within a single decision loop.

B Complementary Feedback and Certificate Derivations

B.1 CDRL Pseudocode

Algorithm 1 presents the general CDRL framework. Certificate generation is optional and independent of the candidate's reward or validity status. In the physics application, certificates arise from the structural-failure and symmetry-redundancy conditions described in Appendix B.3.

[2]In a complete model, the VEV is a dynamical result of the theory, and achieving a certain VEV pattern is a problem in its own right. Here, as in Ref (Baretz et al., 2025), we do not attempt to solve this problem, but assume that a certain set of VEVs can be achieved

Algorithm 1 Certification-Driven Reinforcement Learning (CDRL)

Notation: π_θ policy-value network; V verifier (the physics evaluation pipeline); $\mathcal{A}$ certificate analyzer; $\mathcal{K}_0, \mathcal{K}$ initial and accumulated constraint sets; H space of candidate hypotheses (models); $h \in H$ one candidate; $\mathcal{H}_{\text{valid}}$ set of discovered valid hypotheses; r the evaluation reward; σ an optional failure or redundancy certificate, with $\sigma = \varnothing$ when none is available; c the constraint derived from σ.

Require: Policy π_θ, verifier V, certificate analyzer $\mathcal{A}$, initial constraints $\mathcal{K}_0$
Ensure: Set of valid theories $\mathcal{H}_{\text{valid}}$

$\mathcal{K} \leftarrow \mathcal{K}_0$ ▷ initialize with known domain constraints
$\mathcal{H}_{\text{valid}} \leftarrow \emptyset$ ▷ store discovered valid solutions
for each training episode **do**
  $h \sim \textsc{Plan}(\pi_\theta, \mathcal{K})$ ▷ construct candidate while respecting accumulated constraints
  $(status, r, \sigma) \leftarrow V(h)$ ▷ evaluate candidate with optional failure or redundancy certificate
  Store trajectory and reward r in replay buffer
  if $status =$ valid **then**
    $\mathcal{H}_{\text{valid}} \leftarrow \mathcal{H}_{\text{valid}} \cup \{h\}$ ▷ record valid hypothesis
  end if
  if $\sigma \neq \varnothing$ **then**
    $c \leftarrow \mathcal{A}(\sigma)$ ▷ convert applicable certificate into reusable constraint
    $\mathcal{K} \leftarrow \mathcal{K} \cup \{c\}$ ▷ exclude the certified invalid or redundant pattern
  end if
  Periodically update π_θ from replay-buffer mini-batches
end for

B.2 Scalar Rewards and Certificate-Based Pruning

During the MCTS rollouts that follow (Silver et al., 2016; 2017), this symbolic reasoning layer is implemented via Boolean Constraint Propagation (BCP) (Davis et al., 1962; Biere et al., 2009), which enforces the accumulated clauses as the agent builds new candidates, blocking exploration of any partial assignment that repeats a previously seen conflicting decision pattern. This differs from a negative reward in *kind*, not degree. A negative reward is a scalar that adjusts the network's weights by a small gradient step. It lowers the *probability* of a decision but never rules it out, so the agent can, and empirically does, revisit the same dead end many times, and the penalty says nothing about *which* of the decisions was to blame. A conflict clause instead operates on the search space itself. It is a hard logical constraint naming the exact assignments responsible, and after a worker imports the clause, BCP rejects partial assignments containing that pattern, with no dependence on how well the network has been trained. In effect, a scalar reward changes where the agent *prefers* to go, whereas a certificate changes where it *can* go. As more certificates accumulate, the feasible search space is progressively pruned, allowing the agent to avoid rediscovering equivalent failure modes and focus expensive physics evaluations on regions more likely to contain valid theories.

As a concrete example of this structured feedback in our setting, consider a candidate neutrino model whose non-Abelian A_4 representation and VEV assignment is structurally rank-deficient for *any* choice of $\mathbb{Z}_N$ charge (Figure 1). Rather than emitting only a scalar penalty for this one candidate, our system extracts a *certificate* identifying the representation pattern responsible and compiles it into a logical constraint that blocks that pattern across the whole charge sector, so it is masked the moment it reappears elsewhere in the search tree, before any physics evaluation is run on it (worked out in detail in Section 4). Concretely, we realize this loop with well-established tools: candidate models are encoded as a Boolean satisfiability (SAT) formula in conjunctive normal form; certificates become *conflict clauses* that are enforced during search by Boolean Constraint Propagation (BCP) (Davis et al., 1962; Biere et al., 2009), the same unit-propagation mechanism used inside modern SAT solvers (Biere et al., 2009); and the search itself is an AlphaZero-style Monte Carlo Tree Search (MCTS) (Silver et al., 2016; 2017) guided by a policy-value network. These three components play distinct, complementary roles at every decision. Neural guidance *proposes* the next assignment, symbolic reasoning *prunes* the assignments a certificate has ruled out, and MCTS *balances* exploration against exploitation over what remains. We quantify the payoff of this structured feedback empirically across three neutrino theory spaces (Section 5), and further show

that the constraints and decision patterns CDRL accumulates expose reusable structure in the theory space (Section 5.4).

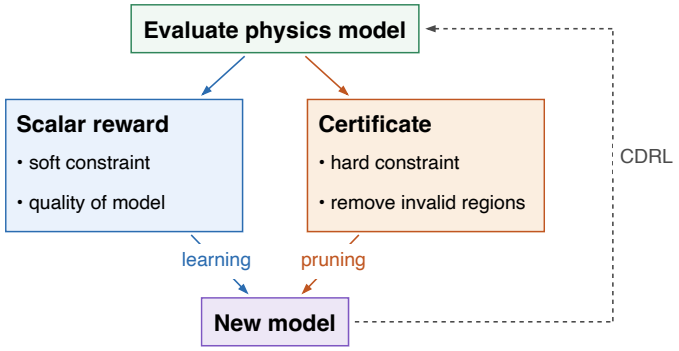


Figure 2: **The two feedback signals CDRL gets from each physics evaluation are complementary rather than competing.** Evaluating a candidate model yields a *scalar reward* that scores its quality and, when an applicable explanation is available, a *certificate*, a *hard* constraint that names the exact components responsible for a failure and forbids them via a clause that Boolean Constraint Propagation (BCP) then enforces everywhere that pattern would recur. The reward shapes the network through *learning*, while the certificate *prunes* the search space. Together they determine the new model the search proposes next, and the cycle repeats. Section 3 gives the formal definition of a certificate.

B.3 Certificate Construction and Soundness

We now give the construction and soundness argument for each certificate type. The soundness condition is that every complete model excluded by a learned clause must either be invalid or be physically equivalent to a model that remains represented in the search. The certificates distinguish two cases: a representation-and-VEV pattern that is invalid under every Abelian charge completion, or a charge assignment that is equivalent to another retained model. Reduction and embedding are two directions of the same symmetry relation and are grouped together in Table 4.

Insufficient-rank representation certificates. The first class arises when a candidate model's non-Abelian representations and VEV choice cannot support sufficiently high-rank mass matrices under any Abelian charge assignment. After constructing a candidate matrix, the system evaluates the model using the physics validity checker. If the model fails, the algorithm performs a reduced analysis in which the Abelian charge sector is temporarily wildcarded and only the non-Abelian representation and VEV assignments are retained. This has the effect of removing the Abelian symmetry entirely.

The particle content and the maximum allowed dimension of the mass terms, fixed here to five, determine the possible mass terms. Symmetries eliminate terms that are not invariant under their transformation laws. Adding a $\mathbb{Z}_N$ symmetry imposes an additional invariance condition. It can remove mass terms, but cannot restore terms already excluded by the non-Abelian symmetry. The representation certificates concern patterns with insufficient mass terms to produce sufficiently high-rank mass matrices before the Abelian symmetry is imposed. No Abelian charge assignment can supply the missing terms. The system extracts the corresponding representation and VEV assignments and generates a conflict clause blocking this combination. Other VEV choices for the same representations remain unaffected and available for exploration. This allows the solver to prune large classes of models that share the same insufficient-rank structure.

Charge reduction and embedding. The second class arises from symmetry relations in the Abelian $\mathbb{Z}_N$ charge sector, independent of the non-Abelian representations or VEV choice. The first relation is reducibility. If the charges and group order share a common divisor d, so that

$$q_i = da_i, \qquad N = dM,$$

then the selection rule

$$q_1 + q_2 + q_3 = 0 \pmod{N}$$

is equivalent to

$$a_1 + a_2 + a_3 = 0 \pmod{M}.$$

The two conditions select the same invariant terms in the Lagrangian, and therefore the same mass matrices and physical predictions. The larger symmetry provides no additional structure beyond the reduced group $\mathbb{Z}_M$. The same reasoning applies in the other direction. A $\mathbb{Z}_M$ model can be embedded into a $\mathbb{Z}_{dM}$ model by multiplying every charge by d. Whichever model is evaluated first, the certificate blocks the other, and the relation is checked in both directions. These clauses concern redundancy with a retained equivalent model, rather than invalidity.

Charge negation. The second equivalence is relation by an automorphism, specifically the order-2 automorphism of negation. Replacing every charge with its additive inverse, $q \mapsto -q \bmod N$, preserves the selection rule $\sum_i q_i \equiv 0 \pmod{N}$: negating every term preserves a sum of zero. The same set of Lagrangian terms remains invariant, so the negated charge assignment describes the same physics and need not be explored separately. Higher-order automorphisms could also be included in future applications.

Clause construction and examples. The system generates a conflict clause by reconstructing the equivalent or excluded pattern in matrix form and negating the literals corresponding to that assignment. For an insufficient-rank certificate, the clause blocks the responsible representation-and-VEV combination independently of Abelian charges. For a symmetry certificate, it blocks the redundant charge assignment. Table 4 illustrates these cases on a common $\mathbb{Z}_6$ candidate: $\mathbb{Z}_6[2, 4, 6]$ reduces to $\mathbb{Z}_3[1, 2, 3]$, while charge negation gives $\mathbb{Z}_6[4, 2, 6]$.

Table 4: **The three certificate types**, each worked through on a common $\mathbb{Z}_6$ candidate; the full derivation is given in Appendix B.3. Certificates 2 and 3 depend only on the $\mathbb{Z}_N$ charges, independent of representations or VEVs. Every clause is ¬(the equivalent or excluded pattern); workers import it from the shared database and enforce it during subsequent search to avoid evaluating matching candidates.

Description	Example	Clause
Insufficient-rank representation Model already rank-deficient without $\mathbb{Z}_N$ symmetry	Reps and VEVs with $\mathbb{Z}_6$ charges wildcarded already rank-deficient	¬(reps & VEV, any charges)
Abelian symmetry reducibility $\mathbb{Z}_N[q_i] \cong \mathbb{Z}_{N/d}[q_i/d]$	$\mathbb{Z}_6\,[2, 4, 6] \mapsto \mathbb{Z}_3\,[1, 2, 3]$	¬(other representative)
Charge negation $\mathbb{Z}_N[q_i] \cong \mathbb{Z}_N[-q_i \bmod \mathrm{N}]$	$\mathbb{Z}_6\,[2, 4, 6] \mapsto \mathbb{Z}_6\,[4, 2, 6]$	¬(negated charges)

B.4 Learned Clause Injection

The conflict clauses generated during evaluation are accumulated globally and injected into the SAT solver before each new search episode, requiring no recompilation since each clause is already in CNF. Detection then falls out of BCP itself. At every MCTS node the current partial assignment is fed to the solver, and to score a candidate block assignment we tentatively add its literals and propagate. If propagation drives any learned clause to a conflict, that block assignment is removed from the legal-action mask before the policy network ever ranks it (Section 4.2). Instead, a forced literal implied by the clauses is propagated into the state. Figure 1 traces this loop end to end. A conflict at a completed model is analyzed into a clause, appended to the shared database, and the clause then masks the same failing pattern the moment it recurs elsewhere in the tree, before any physics evaluation is spent on it. This is precisely where the neuro-symbolic design pays off over a purely neural agent. The learned clause makes the excluded assignment pattern logically unreachable

after each worker imports the clause (Figure 2), so large regions of the search space are pruned permanently rather than merely down-weighted.

B.5 Soundness Scope and Failure Predicate

The representation-level failure considered here is insufficient mass-matrix rank for the fixed non-Abelian representations and VEVs before an Abelian symmetry is imposed. Its certificate concerns all Abelian completions of that pattern. The soundness argument therefore requires structural rank insufficiency before the Abelian symmetry is imposed; it does not justify blocking a representation-and-VEV pattern solely because a completed model has a poor fit or exceeds the parameter-count cutoff. These terminal-evaluation criteria are distinct from the reduced rank predicate used to justify this certificate. In contrast, charge reduction, embedding, and negation establish equivalence to a retained model; they do not establish that the excluded model is invalid. The construction and scope of these two classes are described in Appendix B.3.

The physics pipeline constructs mass matrices, checks their rank, fits the observables, and determines validity independently of the learned policy. The certificate analyzer applies fixed rules to identify the assignments responsible for structural failure or symmetry redundancy. Neither the physics pipeline nor the analyzer depends on policy weights, value estimates, or MCTS visit counts. The agent chooses which candidates to propose, but does not change the rules used to check them. These certificates are distinct from the post-hoc KD rules, which summarize search preferences and are reused as soft guidance rather than as validity criteria.

The full theory spaces exceed 10^{26} candidates; the certificate arguments above describe the structural-failure and symmetry-equivalence cases instead of enumerating the full space.

C Model Representation and Sequential Construction

C.1 Matrix Representation of Candidate Models

Following the AMBer framework (Baretz et al., 2025), we encode a candidate theory as a matrix representation that captures the assignment of particles to symmetry representations and charges. Rows correspond to particles in the model, while columns correspond to irreducible representation and charge assignments under the symmetry groups. For example, rows include the three charged lepton doublets (L_1, L_2, L_3), charged lepton singlets (E_1, E_2, E_3), right-handed neutrinos (N_1, N_2, N_3), Higgs fields (H_u, H_d), and several flavon fields. Columns represent discrete non-Abelian representations (e.g., A_4 irreducible representations), Abelian $\mathbb{Z}_N$ charges, and VEV configurations for flavon fields. Columns are encoded using one-hot vectors. Unassigned entries are represented by -1, allowing the state to represent partially constructed models. This matrix therefore forms the state representation for the learning agent.

C.2 Sequential Construction of Models

Model construction is formulated as a sequential decision process. Instead of assigning every matrix entry independently, decisions are organized into blocks corresponding to logical modeling choices. The agent starts by assigning the active $\mathbb{Z}_N$ symmetry as one block, then for each particle in turn assigns its representation and then its Abelian charge as two separate blocks, and finally assigns each flavon's VEV configuration as its own block. At each step, the next unassigned block is selected, and the agent chooses from a small alphabet consisting of either an all-zero vector (inactive block) or a one-hot vector indicating the assignment. This structured action space reduces the branching factor and enables incremental construction of candidate theories until completion.

D Additional Qualitative Results

Qualitative trends. Aggregate rates measure how often a search succeeds, but for model building the *diversity* of what is found matters just as much. Physicists want a broad set of distinct candidate theories to study, not many copies of the same one, and the distribution across symmetry groups provides one measure of this coverage. We therefore also examine the distribution of discovered

models, not only their counts. Figure 3 shows the clearest evidence of this. CDRL discovers viable models spread across a substantially broader range of $\mathbb{Z}_N$ symmetry groups than AMBer, which concentrates around a few specific values of N. Models are binned by the $\mathbb{Z}_N$ they were constructed under rather than a reduced or canonical form, so this distribution measures coverage of the assigned symmetry groups. Reducibility certificates (Table 4) target equivalent charge assignments to reduce repeated exploration of redundant representatives. Figure 4 shows where CDRL's models concentrate in (χ^2, n_p) space in the right column compared to AMBer in the left column. We also present a strong example model discovered by CDRL in the $T_{19} \times \mathbb{Z}_4$ space (Table 5). CDRL contributes an additional set of viable neutrino models beyond those reported by AMBer, evaluated with the same physics pipeline and viability criteria. Together with the broader $\mathbb{Z}_N$ coverage in Figure 3, these discoveries support complementary exploration of the theory space and broaden the set of candidate models available for further physics study.

Following the analysis protocol of AMBer (Baretz et al., 2025), we compare these distributions in more detail. For the distribution of models shown in Figure 4, CDRL reaches neutrino models with the smallest parameter counts observed for either method (Table 1 reports this as Min Params) while maintaining competitive or lower χ^2 values. For the $\mathbb{Z}_N$ distribution of Figure 3, AMBer concentrates its discoveries around specific symmetry groups (e.g., $N = 5$), whereas CDRL discovers viable models across a broader range of $\mathbb{Z}_N$ choices. As the learned clause database grows, the legal-action mask becomes increasingly selective, allowing Boolean constraint propagation to eliminate larger regions of invalid or redundant search trajectories before expensive physics evaluation is performed. Reducing these repeated evaluations makes more of the evaluation budget available for distinct candidates. Neural guidance, MCTS, and shared certificates jointly determine which candidates are explored. CDRL's reward contains no term that rewards higher-order $\mathbb{Z}_N$ symmetries (Section 4.3 gives the full reward, which depends only on χ^2 fit quality and the number of free parameters); AMBer's scalar reward includes such a term to steer search toward the more complex, higher-order groups (Baretz et al., 2025). The broader coverage observed for CDRL is therefore achieved without an equivalent reward incentive. This difference in reward design does not affect the evaluation protocol itself (Section 5), which scores both methods identically.

Table 5: **Example neutrino flavor model discovered by CDRL in the $T_{19} \times \mathbb{Z}_4$ theory space.** The table shows particle representation and charge assignments for one of the viable neutrino models discovered by CDRL. This model achieves a strong χ^2 fit with a minimal number of free parameters, illustrating the ability of CDRL to discover compact and high-quality theories. Additional discovered neutrino models across all three theory spaces will be included in the supplementary materials.

	L	E_1	E_2	E_3	N	H_u	H_d	ϕ_1	ϕ_2	ϕ_3	ϕ_4	ϕ_5	ϕ_6
T_{19}	$\mathbf{3}_1$	$\mathbf{1}''$	$\mathbf{1}$	$\mathbf{1}'$	$\bar{\mathbf{3}}_2$	$\mathbf{1}'$	$\mathbf{1}$	$\mathbf{3}_1$	$\bar{\mathbf{3}}_1$	$\mathbf{3}_1$	$\bar{\mathbf{3}}_1$	$\mathbf{3}_2$	$\mathbf{3}_2$
$\mathbb{Z}_4$	4	2	2	2	4	2	4	2	2	3	4	4	1

E Search Knowledge Discovery: Extraction and Examples

We describe the procedure used to extract soft constraints from MCTS trajectories in Algorithm 2 and incorporate them into the search process. We retain high-confidence decision points from MCTS trajectories, *i.e.* states where the dominant action receives more than 95% of visits and has a positive value estimate, and group these states by construction progress, defined as the number of assigned entries in the state before the first unassigned position. For each progress bucket with enough samples, we train one shallow decision tree that predicts the dominant action from the partial model's earlier assignments; hyperparameters are given in Appendix G. Because blocks are assigned in a fixed top-down order (Appendix C), only blocks assigned earlier in that order are available as features. We extract these high-confidence root-to-leaf paths as rules of the form conditions $\rightarrow a$, where the conditions are conjunctions over binary state features, and retain only rules expressible as binary conditions for compatibility with the symbolic representation. These rules can then be injected into a new CDRL run as soft constraints.

At inference time, we identify all rules whose conditions are satisfied by the current state, let each activated rule vote for its preferred action to form a bias vector $r(s)$, and reweight the neural policy as $\tilde{P}(a \mid s) \propto P(a \mid s)\,(1 + \beta\, r_a(s))$, where $\beta = 0.5$ in all experiments. The reweighted policy is then

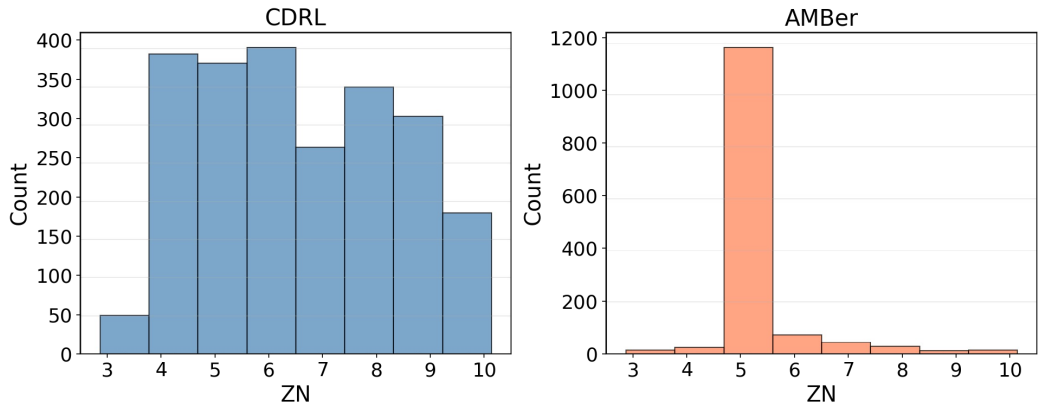


Figure 3: Distribution of successfully discovered neutrino models across $\mathbb{Z}_N$ symmetry groups for AMBer and CDRL. In these experiments, CDRL discovers viable models across a broader range of symmetry choices, while AMBer concentrates more heavily around specific values of N. Models are grouped by their assigned $\mathbb{Z}_N$.

masked by the legality constraints and renormalized before being used in MCTS, so soft constraints guide the search without overriding hard constraints or eliminating exploration.

Algorithm 2 Soft Constraint Extraction and Policy Reweighting

Require: MCTS traces $\mathcal{D}$, progress buckets $\mathcal{B}$
Require: Minimum samples m, tree hyperparameters (d, s, ℓ)
Require: Soft bias strength β
Collect state-action samples (x_i, a_i) from MCTS
Retain high-confidence samples based on visit concentration and value estimates
Group samples into buckets $\{\mathcal{D}_b\}_{b\in\mathcal{B}}$ based on progress
for each bucket $b \in \mathcal{B}$ **do**
    if $|\mathcal{D}_b| < m$ **then**
        continue
    end if
    Train decision tree T_b with depth d, split s, leaf size ℓ
    Extract leaf paths corresponding to high-confidence predictions
    Convert each path into a rule (conditions $\rightarrow a$)
    Store rules with support and purity
end for
Build rule table indexed by progress and decision block

Policy Reweighting during MCTS:
Given state s, compute neural policy $P(\cdot \mid s)$
Identify rules satisfied by s and compute bias vector $r(s)$
Reweight policy:

$$\tilde{P}(a \mid s) \propto P(a \mid s)\,(1 + \beta\, r_a(s))$$

Apply legality mask and renormalize $\tilde{P}$

E.1 Rule Interpretation and Limitations

The extracted rules are simple if-else conditions over particle assignments that reveal recurring structure in successful models. For example, assignments of charged lepton singlets often follow consistent patterns (e.g., E_1 influencing E_2, E_3 through identical or systematically shifted $\mathbb{Z}_4$ charges). We also observe coordinated triplet assignments across L, N, and ϕ, mirroring common design

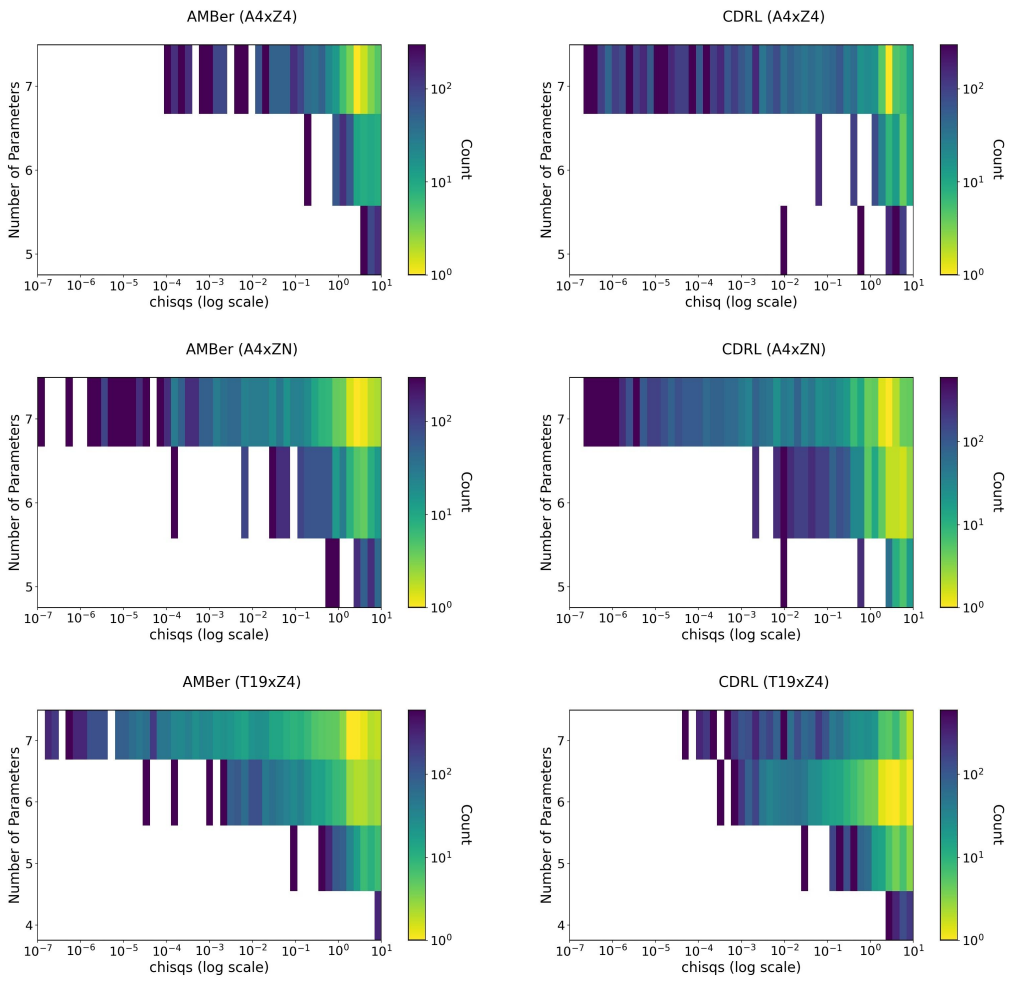


Figure 4: Distribution of discovered models over χ^2 (fit quality) and n_p (number of parameters). Each bin shows model density, with the bottom-left region corresponding to well-fitting, low-complexity models. CDRL places more mass in the low-n_p areas. Within that region, χ^2 differences at very low values do not indicate meaningfully different fit quality, so Table 1 and Table 3 remain the primary quantitative comparison, with this figure illustrating where in (χ^2, n_p) space that advantage is concentrated.

patterns used by human model builders (Ding & Meloni, 2012). Although extracted post-hoc, these rules provide reusable and generalizable structure. The full set of 40 rules is listed in Appendix H.

Crucially, interpreting each rule within the model building process requires additional context not always captured within the decision tree, and many rules are more representative of the agent's search process than of correlations between parameters of the model. For example, several of the observed rules for the $\mathbb{Z}_N$ charges of the charged lepton singlets (E_1, E_2, E_3) show them favoring identical or closely related charges. This is a reasonable pattern for correctly producing their masses if other particles have already been given appropriate representation assignments. However, some trees predict only a subset of the charge values the symmetry allows, favoring those values when reused as soft guidance. This demonstrates the limitation of interpreting these rules physically. Each tree is fit only to the high-confidence trajectories within a single progress bucket, a small, selective sample. Therefore, a tree can look like a high-confidence rule simply because the values outside it were rare in that particular sample, not because the wider search avoids them. While we introduce a tool for search knowledge discovery, further work in this direction may enable more robust interpretation of these discovered correlations in terms of useful model-building rules. Surfacing this kind of observation is still useful. It either flags a rule that will not hold up once tested on more data, or, when a pattern

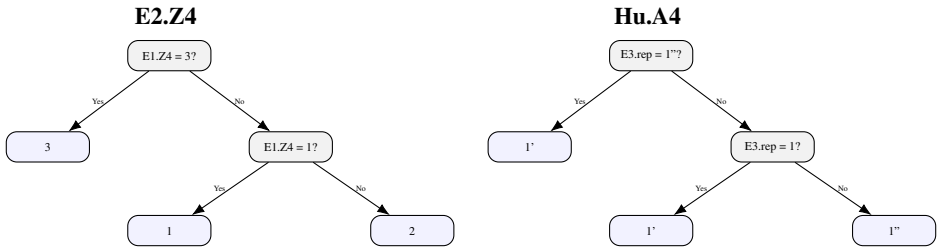


Figure 5: **Example decision trees.** *Left:* Determination of the E_2's $\mathbb{Z}_4$ charge in $A_4 \times \mathbb{Z}_4$. Among high-confidence states, this decision tree predicts the E_2's $\mathbb{Z}_4$ charge from E_1's $\mathbb{Z}_4$ charge. This demonstrates how CDRL has learned to match the $\mathbb{Z}_4$ charges of the charged lepton singlets, a reasonable result in order to construct physically realizable mass matrices. This tree predicts a subset of the possible choices, omitting the $\mathbb{Z}_4$ charge value of 4. When reused as soft guidance, it favors these choices without prohibiting other legal charge values. *Right:* Determination of the H_u's A_4 representation in $A_4 \times \mathbb{Z}_N$. This decision tree is harder to interpret physically. The up-flavor Higgs is not directly correlated with the charged lepton sector, so there is either some non-trivial relationship, or this is a spurious correlation within the CDRL search. The full set of such observations for all three theory spaces is in Appendix H.

holds up under closer inspection, a regularity of the theory space a human model builder had not already anticipated.

F The CDRL Loop: A Step-by-Step Walkthrough

Figure 7 below walks through a worked example as four snapshots of the same tree, taken as it is actually built, covering the same components as Figure 1 (the policy-value network, the BCP referee, MCTS, physics evaluation, the certificate analyzer, and the clause database) except search knowledge discovery, which operates offline after training and is not part of this per-candidate loop. Showing the same tree across panels also makes the key claim concrete rather than only described. A clause learned from one branch permanently prunes a matching pattern anywhere else in the tree.

At every node, not only the root, the BCP referee first masks any child that would violate an already-learned clause, and the policy-value network then ranks the remaining legal children. This repeats at each subsequent decision (the $\mathbb{Z}_N$ choice, then each particle's representation, then its charge, and so on). What Figure 7 shows as a single selected path is itself the outcome of many such simulations from the current state, each one descending through already-expanded nodes by upper-confidence bound and expanding one new leaf. The move actually played is the one visited most often across these simulations, not simply the network's top-ranked action. As in Algorithm 1, the network is updated periodically from replay-buffer mini-batches rather than immediately after every single evaluation, and Step 3 illustrates the certificate path for an applicable structural failure. Symmetry certificates can also identify redundancy in otherwise valid candidates. The three $\mathbb{Z}_N$ choices at the root are an illustrative subset of the nine allowed in this theory space ($N \in \{2, \ldots, 10\}$).

G Additional Experimental Details

G.1 Compute Infrastructure

All CDRL experiments are conducted on a machine with Intel Xeon Gold 6448Y CPUs (32 cores, 2.10 GHz), 32 GB RAM, and an NVIDIA H100 GPU with a 20 GB MIG partition. Parallel self-play is executed across CPU workers, while neural-network training is performed on the GPU.

The dominant computational cost in our setting arises from the physics evaluation pipeline, which involves symbolic model construction, numerical optimization, and parameter fitting for candidate theories. Both methods execute this pipeline on CPUs. CDRL additionally uses a GPU partition for neural-network training, while AMBer runs entirely on CPUs. Table 6 places the number of discovered neutrino models alongside elapsed time and resource usage. The terminal-evaluation budgets are reported separately in Table 2.

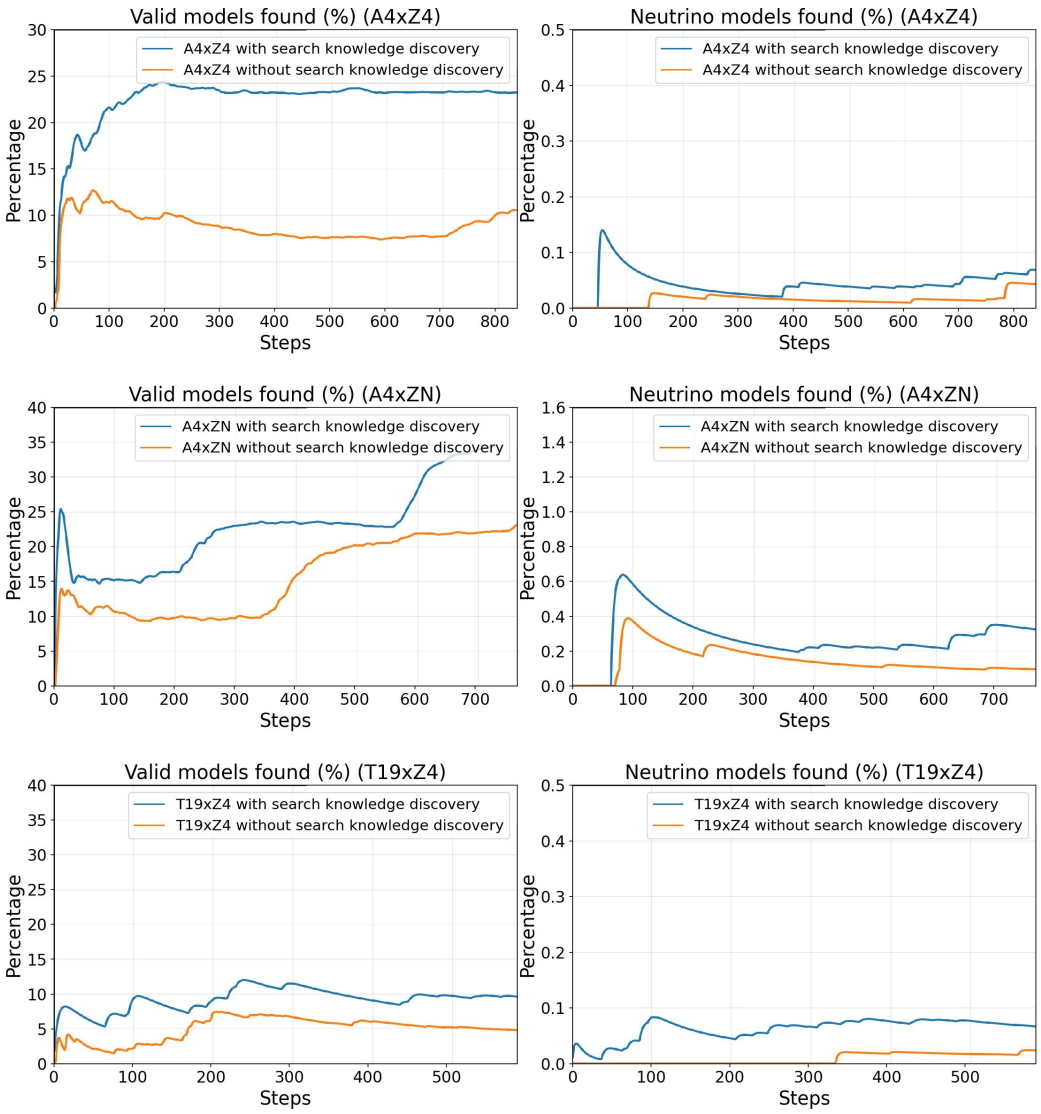


Figure 6: Impact of search knowledge discovery (KD) across theory spaces. Each panel is titled by metric and theory space. We compare CDRL with and without rule-based soft constraints on $A_4 \times \mathbb{Z}_4$, $A_4 \times \mathbb{Z}_N$, and $T_{19} \times \mathbb{Z}_4$. Incorporating rules extracted from high-confidence search trajectories consistently improves both valid model rates and neutrino model discovery. Rule-guided runs achieve higher final performance and faster convergence, with gains of roughly 1.5–2$\times$ in valid models and 2–3$\times$ in neutrino models across the three spaces, demonstrating that the extracted rules provide useful and generalizable guidance for exploration. These experiments are performed using smaller-scale KD evaluation runs, so absolute values are not directly comparable to the full-scale results reported in Table 1.

CDRL discovers more neutrino models with one-quarter of the terminal physics evaluations used by AMBer. This reduction reflects improved search efficiency independently of the speed of neural-network training. It is complemented by lower reported wall-clock time and CPU usage in all three theory spaces (Table 6). CDRL uses 32 portfolios across CPU workers and one shared GPU partition, whereas AMBer uses 250 parallel environments. The timing results describe these respective configurations, while the terminal-evaluation counts quantify the reduction in calls to the physics pipeline.

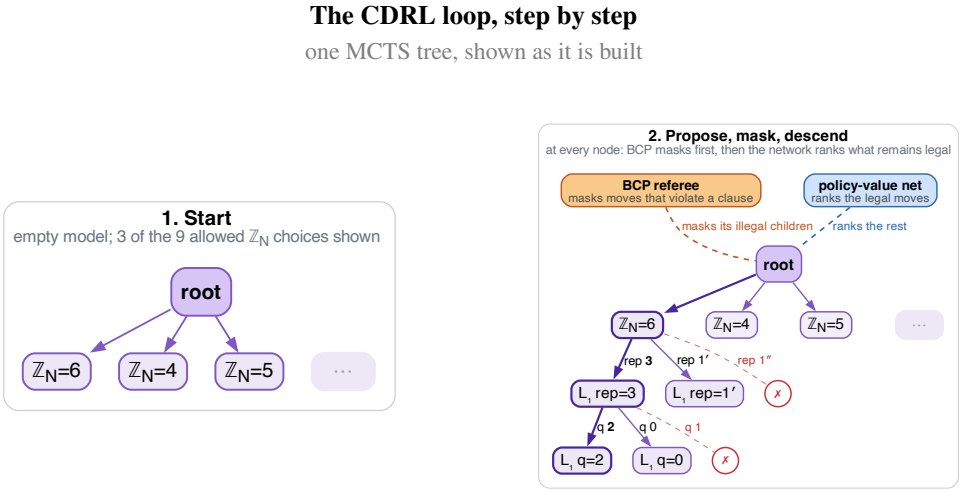


Figure 7: **The CDRL mechanism, shown as four snapshots of the same tree. (1) Start:** the empty model; three of the nine allowed $\mathbb{Z}_N$ choices are shown at the root. **(2) Propose, mask, descend:** at the root, and again at every node MCTS visits, the BCP referee masks the illegal children first and the policy-value network then ranks what remains; MCTS commits to $\mathbb{Z}_N = 6$ and descends, and the same red $\times$ masking recurs at each node along the way. Continued below.

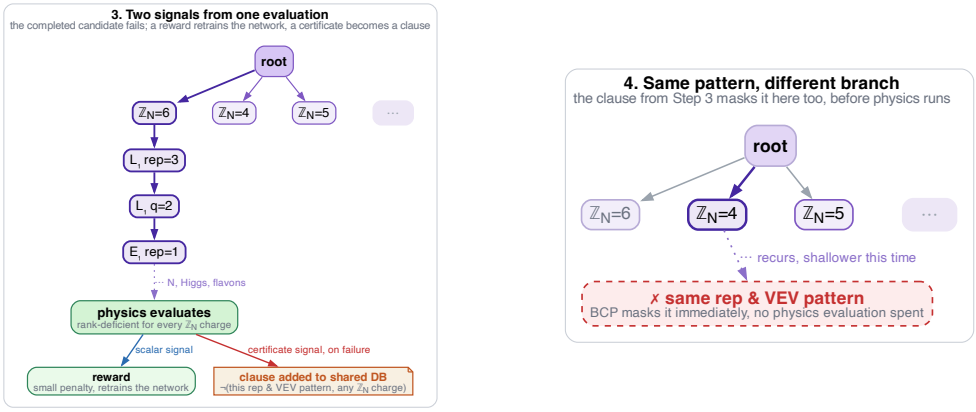


Figure 7: **(continued) (3) Two signals from one evaluation:** the branch completes and the candidate fails, rank-deficient for every $\mathbb{Z}_N$ charge; this single evaluation produces a scalar reward that retrains the network and, separately, a certificate that the analyzer converts into a clause added to the shared database. **(4) Same pattern, different branch:** MCTS next explores the sibling $\mathbb{Z}_N = 4$ branch, and the clause learned in step 3 masks the same representation-and-VEV pattern as soon as it recurs, before the branch is built out further or any physics evaluation is spent on it.

Table 6: **Discovery output and reported compute.** AMBer's model counts are taken from Baretz et al. (2025), and its wall-clock and CPU-hour figures were provided by the authors through personal communication. Both methods' model counts match Table 2. Counts denote inequivalent filtered models for normal ordering under the same filtering and equivalence criteria as AMBer. CPU-hours and GPU-hours are reported separately. †GPU-hours for CDRL denote conservative upper bounds for one H100 20 GB MIG partition, counting the full run duration, including idle periods. AMBer uses no GPU.

Space	Method	Neutrino models found	Wall-clock (h)	CPU-h	GPU-h†
$A_4 \times \mathbb{Z}_4$	AMBer	683	14.7	73.5	0
	CDRL	**1,090**	1.2	23.7	≤ 1.2
$A_4 \times \mathbb{Z}_N$	AMBer	1,394	13.4	67.0	0
	CDRL	**2,343**	1.3	24.1	≤ 1.3
$T_{19} \times \mathbb{Z}_4$	AMBer	6,439	21.7	434.0	0
	CDRL	**7,019**	5.2	148.5	≤ 5.2

The GPU-hour bounds count one shared partition over the entire run, including periods when CPU workers perform physics evaluations and the GPU is idle. Active neural-network training occupies only part of this duration. CDRL uses a compact policy-value network with two fully connected hidden layers of 512 and 256 units. Across all the runs, training uses at most 30 iterations with at most 20 epochs per iteration, and neural-network training takes at most 9 seconds per iteration. The table therefore reports a conservative bound rather than measured active GPU training time. Training barriers and waiting time are already included in the end-to-end wall-clock measurements. CPU-hours and GPU partition-hours are reported separately because they represent different computational resources.

G.2 Additional Implementation Details

Our framework is implemented in PyTorch. We integrate symbolic constraint reasoning using a SAT solver backend, leveraging MiniSAT-style Boolean Constraint Propagation (BCP) via the PySAT library (Ignatiev et al., 2018).

Domain constraints are compiled into conjunctive normal form (CNF) and enforced during search. After each action, the solver applies BCP to propagate implied assignments, ensuring that all intermediate states remain consistent with hard constraints.

For search knowledge discovery rule extraction (Section 5.4), we treat an MCTS decision as high-confidence when the dominant action receives more than 95% of visits and has a positive value estimate. Progress buckets require at least 10 such samples; each decision tree has maximum depth 3, requires at least 10 samples to split a node and 5 samples per leaf, and is fit with an 80/20 train-test split when enough data is available.

Hard constraints, reward, and KD guidance. The initial one-hot and structural rules and the learned conflict clauses are hard Boolean constraints in CNF. They express exact discrete choices, not numerical tolerance bands, and the scalar reward cannot override them. Mass-matrix rank checks, parameter count, and the fitted χ^2 belong to terminal evaluation. The reported neutrino-model thresholds are $\chi^2 \leq 10$ and $n_p \leq 7$. KD rules are extracted after training and, when reused, bias policy preferences without automatically removing alternative legal actions.

Verification of discovered models. The external `Model2Mass/flavorpy` pipeline constructs and evaluates terminal candidates independently of the learned policy, using the NuFit 5.3 observables. As a physics sanity check, particle-physics experts on the team inspected representative viable models and confirmed the consistency of their representation assignments, interaction structure, mass matrices, and fitted observables. These checks complement the established evaluation pipeline, also used by AMBer, and provide additional confidence in the physical consistency and phenomenological viability of the discovered models. Table 5 gives one example, and the discovered models and code will be

included in the supplementary materials for independent inspection and reproduction. The KD rules are provided in Appendix H.

G.3 BASELINES

We compare against **AMBer** (Autonomous Model Builder) (Baretz et al., 2025), an RL framework for neutrino flavor model discovery. AMBer performs iterative model refinement using scalar rewards derived from the physics evaluation pipeline, including χ^2 fit quality, the number of free parameters, and a bonus for exploring higher-order $\mathbb{Z}_N$ symmetry groups. Unlike CDRL, AMBer does not use certification-driven symbolic pruning or reusable constraint learning.

We additionally compare against several ablated and hybrid baselines, including MCTS + symbolic feedback, DNN + symbolic feedback, symbolic-only SAT-based search (Liang et al., 2016), and a DNN+MCTS framework (Silver et al., 2016). These baselines isolate the contributions of planning, neural guidance, and symbolic reasoning.

Controlled ablations. Table 3 uses the same terminal-evaluation budget, physics evaluator, and reward within each theory space. "No symbolic" keeps the DNN and MCTS but disables certificate extraction and propagation of learned clauses. It is the scalar-feedback control. "No DNN" removes the policy-value network while retaining MCTS, certificate extraction, SAT propagation, clause sharing, and terminal caching. It is therefore the complementary control for uninformed planning with symbolic pruning. "No MCTS" removes tree search while retaining the remaining components. "No portfolio" disables cross-worker clause and replay sharing.

For example, full CDRL versus "No symbolic" yields valid/neutrino rates of $28.1/0.10\%$ versus $14.9/0.01\%$ on $A_4 \times \mathbb{Z}_4$, $26.9/0.19\%$ versus $16.2/0.03\%$ on $A_4 \times \mathbb{Z}_N$, and $18.2/0.09\%$ versus $5.4/0.04\%$ on $T_{19} \times \mathbb{Z}_4$. "No DNN" produces no neutrino models at the reported precision in all three spaces. Thus neither symbolic pruning nor neural-guided planning alone accounts for the full performance. Together, these controlled comparisons show that symbolic pruning, learned neural guidance, and MCTS planning each contribute to the full system's performance.

G.3.1 LLM BASELINE DETAILS

We evaluate a prompting-based baseline using GPT-5.3 (Singh et al., 2025). The model is prompted to directly generate candidate neutrino flavor model matrices satisfying the required representation, charge, and VEV assignment format used throughout the paper.

The prompt contains: (i) a task description explaining the flavor model construction problem, (ii) formatting instructions specifying the matrix structure and allowed assignments, and (iii) 3 few-shot examples of valid candidate model matrices drawn from the corresponding theory space. The model is then asked to generate new candidate matrices autoregressively.

Generated outputs are parsed into matrix form and evaluated using the same symbolic constraints and physics evaluation pipeline used for all other methods. Invalid, incomplete, or unparsable outputs are counted as failed candidates.

We use temperature 0.8 and top-p 0.95 decoding for generation. The baseline does not use search, SAT-based propagation, iterative refinement, or certification-driven feedback, and therefore serves as a reference point for pure prompting-based generation without structured exploration.

G.4 NEURAL NETWORK ARCHITECTURE

The policy-value network operates on a flattened matrix representation of the state. The architecture consists of:

- Fully connected layer: 512 units with BatchNorm and ReLU
- Fully connected layer: 256 units with BatchNorm and ReLU
- Policy head: linear layer to action dimension followed by log-softmax
- Value head: linear layer followed by $\tanh$

G.5 Additional Training Details

We provide the concrete hyperparameters for the MCTS and network training described in Section 4. We use:

- Max iterations: 30
- Episodes per iteration: 100
- MCTS simulations per move: 150
- MCTS c_{puct}: 0.5
- Batch size: 512
- Training epochs per iteration: 20
- Optimizer: Adam

The policy head is trained using cross-entropy loss against MCTS visit counts, while the value head is trained using mean squared error against normalized terminal rewards.

After each training iteration, the updated model is evaluated against the previous checkpoint in a single-agent self-play evaluation setting. Both models independently generate candidate solutions using the same environment and evaluation pipeline, and we compare their cumulative terminal rewards across a fixed number of evaluation episodes. The checkpoint is updated only if the new model achieves higher cumulative reward than the previous model. Unlike standard AlphaZero pit-play, this comparison does not involve a competitive two-player game, since neutrino model discovery is a single-agent environment.

G.6 Parallelization

We employ a portfolio-style parallelization strategy inspired by parallel SAT solvers. The 32 portfolios independently perform MCTS-guided self-play while exchanging:

- neural network replay buffers
- a global database of learned conflict clauses

Conflict clauses discovered by any worker are shared globally, with workers importing shared clause batches. Accumulated clauses are injected into the SAT solver before new search episodes and enforced through BCP. The workers also share a cache of completed physics evaluations to avoid redundant computation. This combination of independent search, clause sharing, and replay sharing improves exploration efficiency and sample efficiency.

The shared database accumulates approximately 3–4 million learned clauses over training. Workers reuse these clauses to reject matching invalid or redundant assignments before terminal physics evaluation. Training logs indicate substantial reuse across runs and theory spaces. Table 3 reports the effect of removing symbolic pruning and portfolio sharing on valid-model and neutrino-model discovery.

Clause accumulation and exploration diversity. Monotonic clause accumulation makes the legal-action mask increasingly selective. Provided that the soundness condition in Appendix B.3 continues to hold for the accumulated clause database, this removes certified-invalid patterns and redundant representatives while preserving at least one representative of each physically distinct valid theory. The resulting restriction reduces repeated exploration of invalid or equivalent models without excluding distinct viable theories. The observed coverage across symmetry groups and complementary discoveries in Appendix D are consistent with this interpretation. Likewise, disabling portfolio sharing lowers discovery performance and increases variance (Table 3). The empirical results complement the soundness argument rather than establish it on their own.

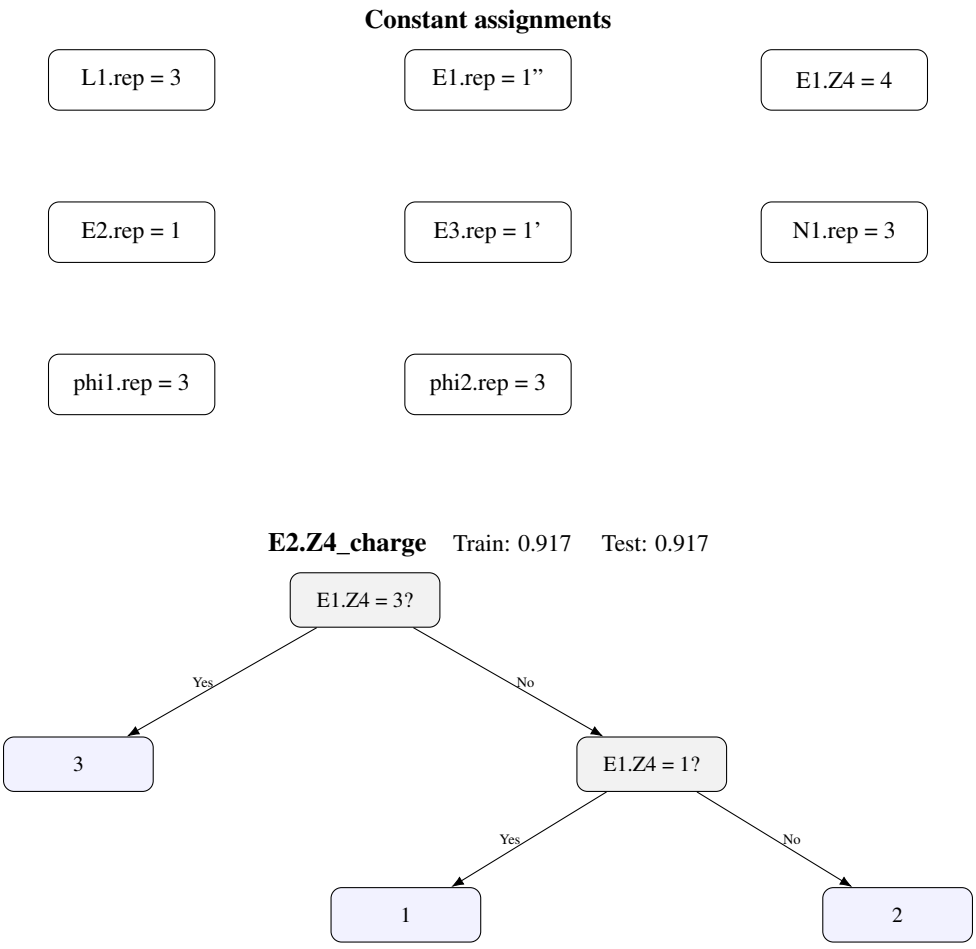
In our current implementation, learned clauses are accumulated monotonically and shared across workers, and we do not employ CDCL-style clause deletion. In practice, this remains computationally manageable because SAT propagation constitutes only a small fraction of the total runtime, while the dominant cost arises from the physics evaluation pipeline. Although the number of learned

certificates grows over training, we did not observe clause management becoming a bottleneck in the theory spaces considered in this work. For larger domains, clause forgetting or activity-based clause management would be a natural extension.

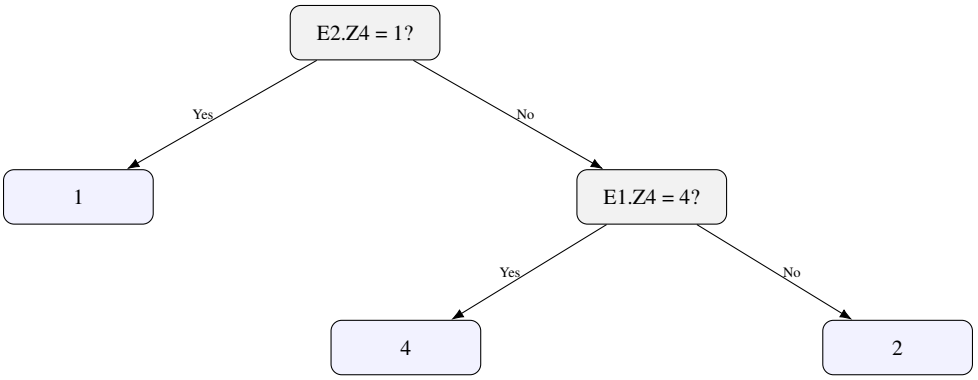
H SEARCH KNOWLEDGE DISCOVERY RULE GALLERIES

Below we list the full set of observations extracted from the high-confidence MCTS trajectories in each theory space, using the procedure described in Section 5.4. Each observation is a shallow decision tree that predicts one block assignment (a representation or a $\mathbb{Z}_N$ charge) from earlier assignments in the same partial model; an observation that collapses to a single leaf is a *constant assignment*, always taking the same value in the high-confidence set. For every tree we report the train and test accuracy of the predicted assignment. In the two-way trees, the left branch is the *Yes* outcome of the condition and the right branch is *No*. These rules are extracted post-hoc and reused as the soft policy bias of Algorithm 2. The aggregate effect of reusing them is reported in Figure 6.

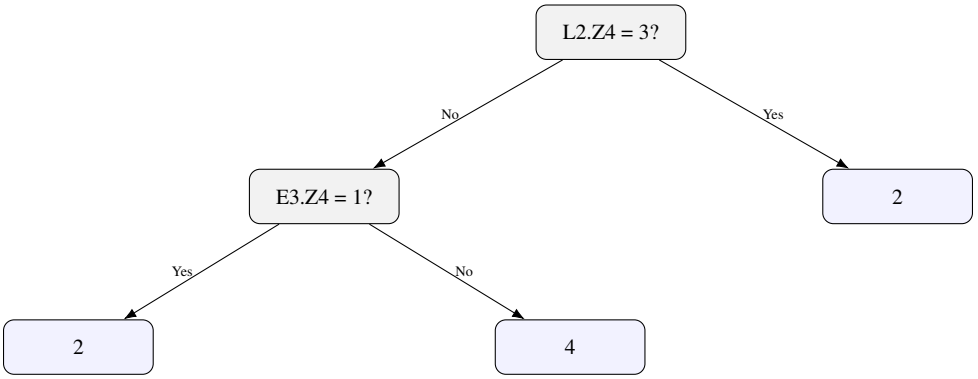
H.1 $A_4 \times \mathbb{Z}_4$



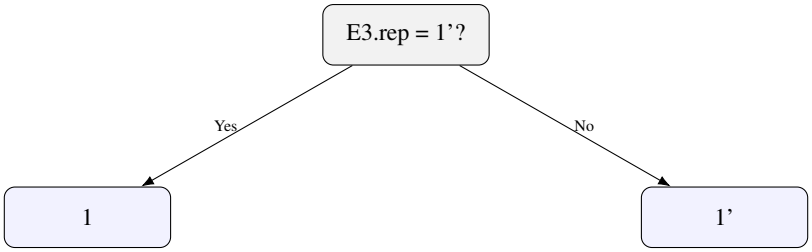
E3.Z4_charge Train: 0.972 Test: 0.944



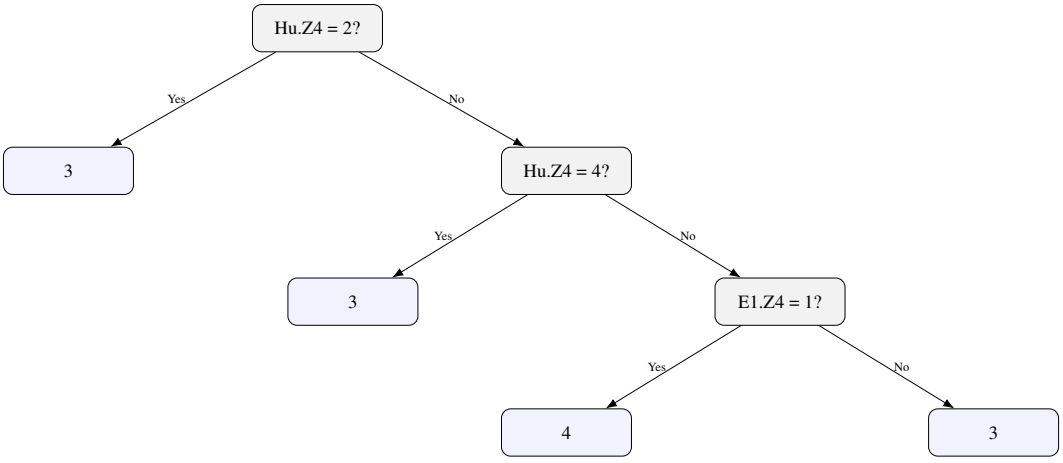
N1.Z4_charge Train: 0.911 Test: 1.000



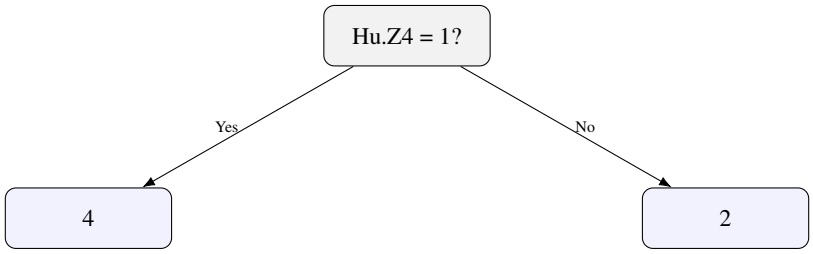
Hd.rep Train: 0.615 Test: NA



phi1.Z4_charge Train: 0.654 Test: 0.429

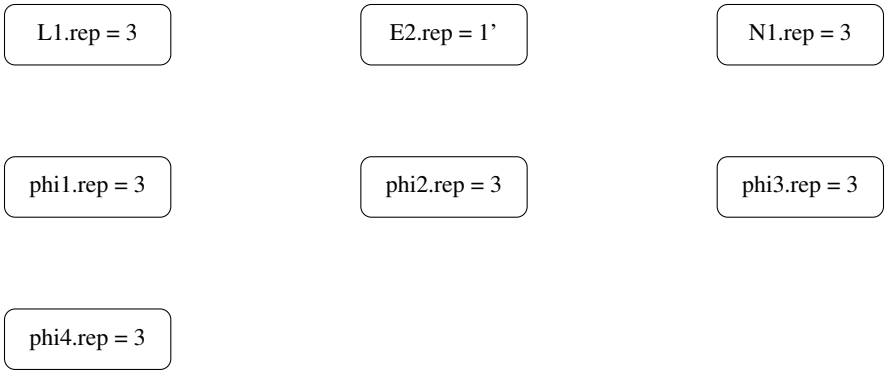


phi2.Z4_charge Train: 0.719 Test: 0.444

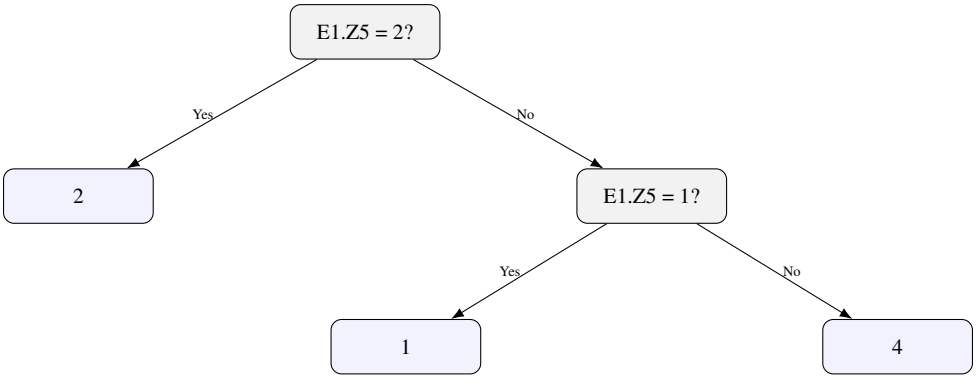


H.2 $A_4 \times \mathbb{Z}_N$

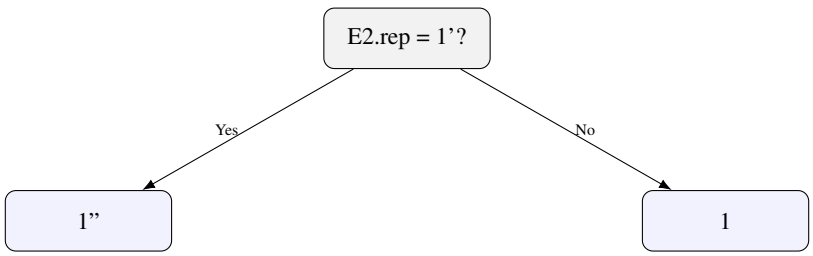
Constant assignments



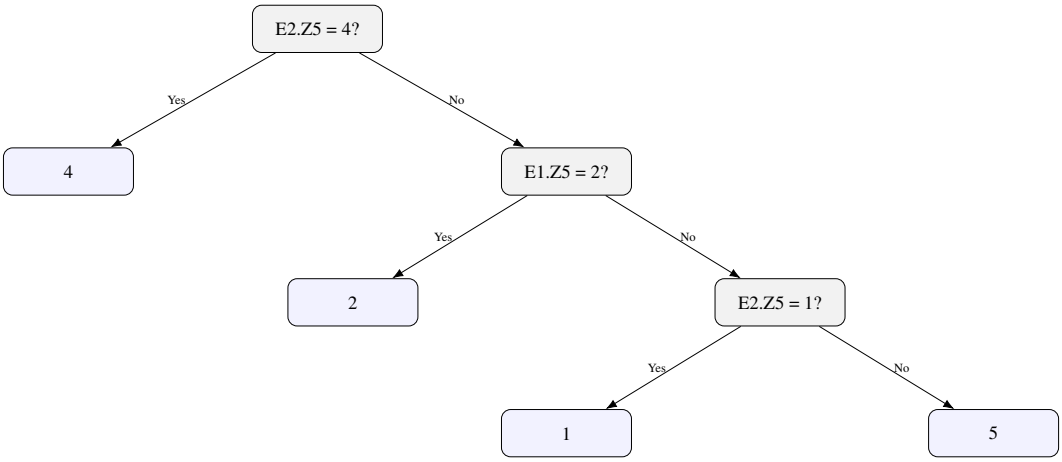
E2.Z5_charge Train: 0.938 Test: 0.882



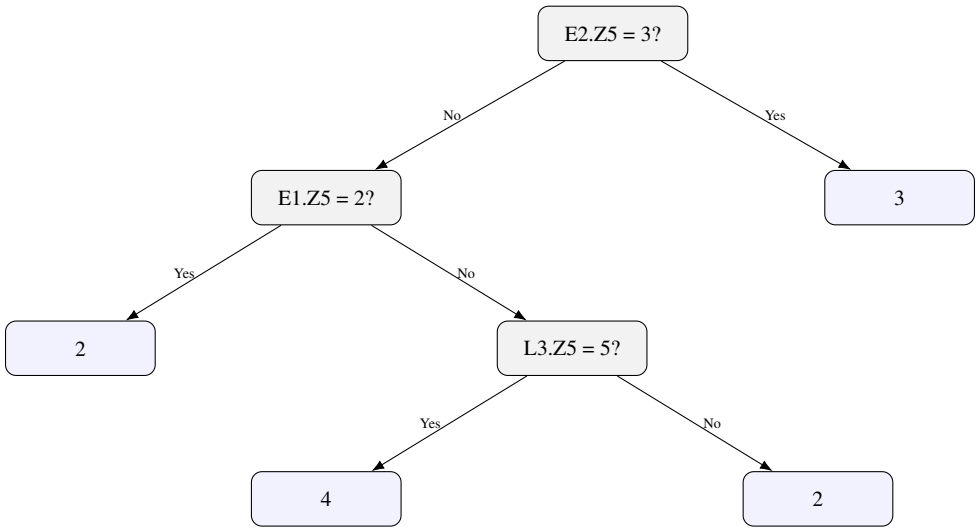
E3.rep Train: 0.707 Test: 0.727



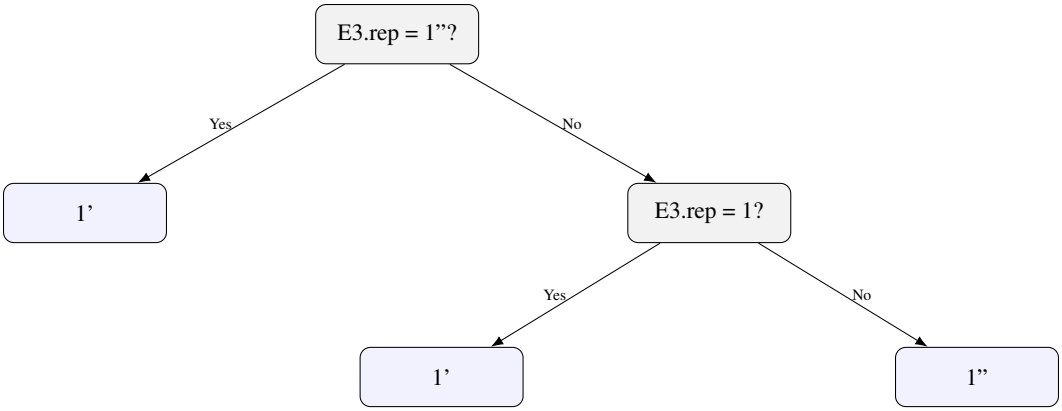
E3.Z5_charge Train: 0.926 Test: 0.968



N1.Z5_charge Train: 0.861 Test: 0.900



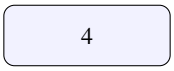
Hu.rep Train: 0.889 Test: 0.600



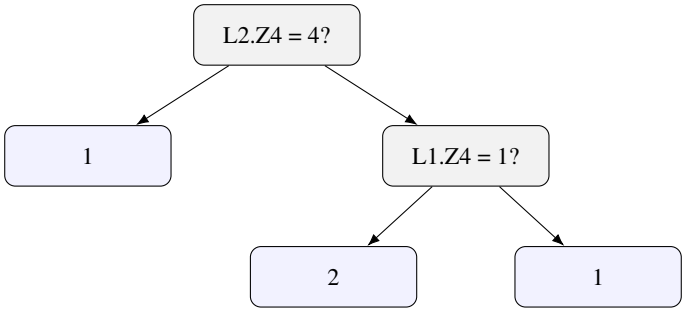
H.3 $T_{19} \times \mathbb{Z}_4$

Left edge: Yes Right edge: No

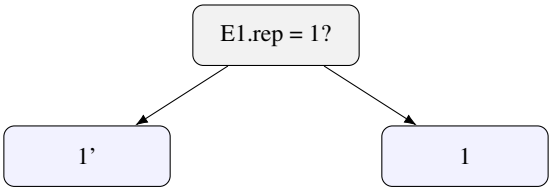
L1.Z4_charge Train: 0.792 Test: 0.714



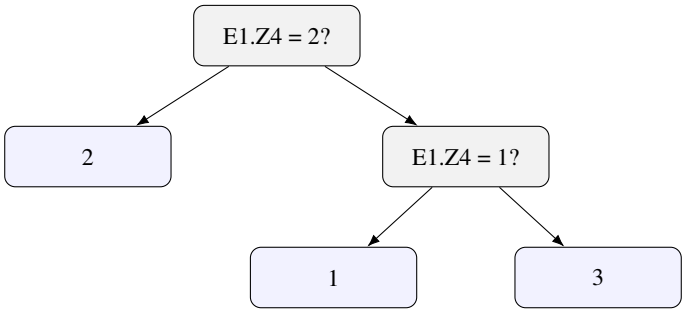
E1.Z4_charge Train: 0.611 Test: 0.600



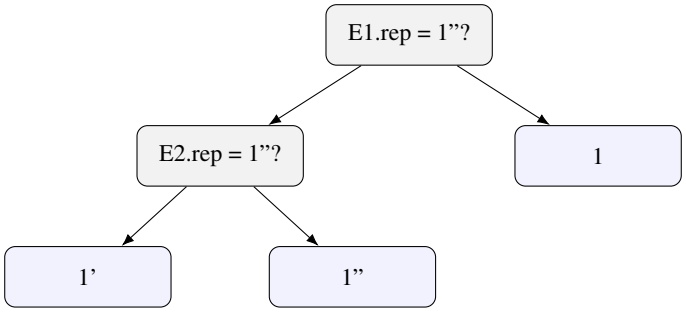
E2.rep Train: 0.950 Test: 0.905



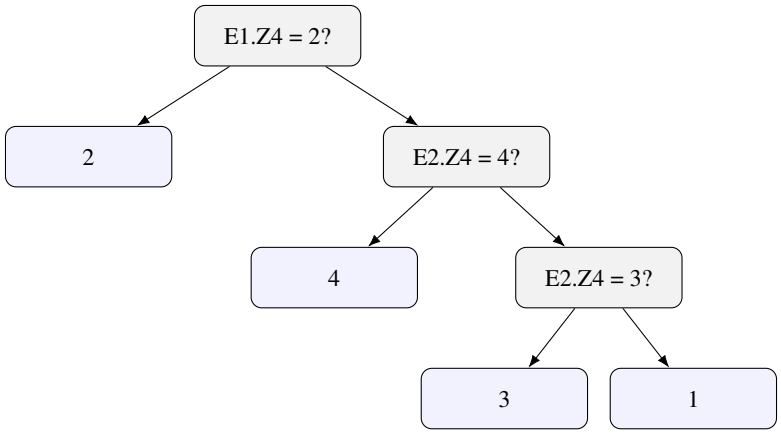
E2.Z4_charge Train: 0.938 Test: 0.857



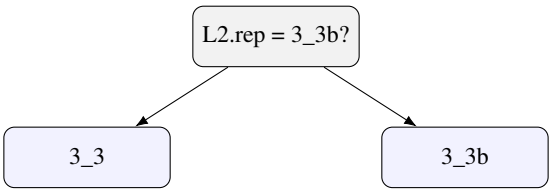
E3.rep Train: 0.685 Test: 0.694



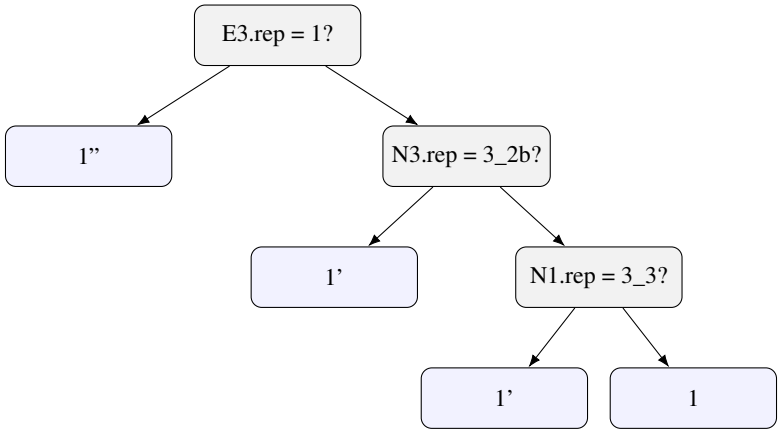
E3.Z4_charge Train: 0.924 Test: 0.926



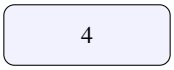
N1.rep Train: 0.960 Test: 0.932



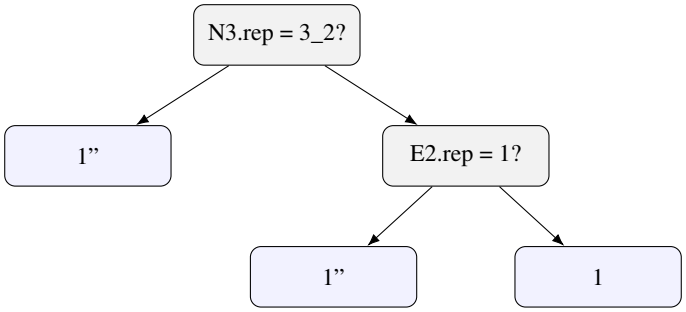
Hu.rep Train: 0.522 Test: 0.667



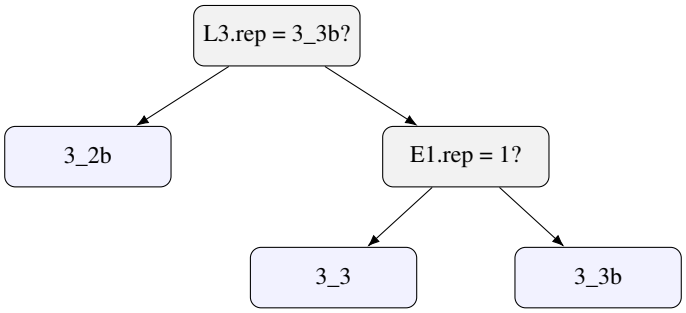
Hu.Z4_charge Train: 0.978 Test: 0.833



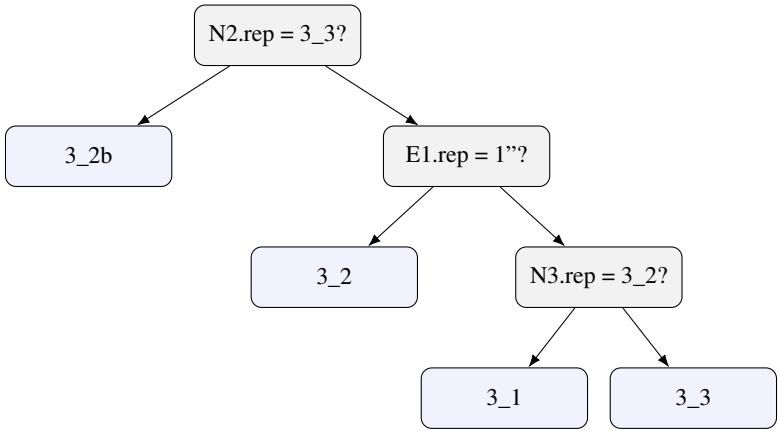
Hd.rep Train: 0.821 Test: 0.875



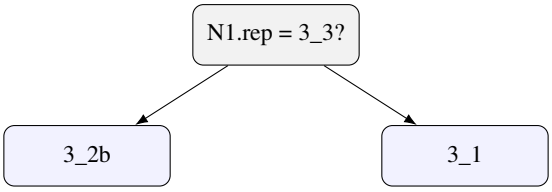
phi1.rep Train: 0.859 Test: 0.941



phi2.rep Train: 0.750 Test: 0.625



phi3.rep Train: 0.741 Test: 0.714



phi5.rep Train: 0.857 Test: 0.750

