THEIA: Learning Complete Kleene Three-Valued Logic in a Pure-Neural Modular Architecture

Augustus Haoyang Li [1]

Abstract

We present THEIA, a modular neural architecture that learns complete Kleene three-valued logic (K3) end-to-end without any external symbolic inference engine, symbolic runtime, or hand-encoded Kleene gate primitives. THEIA processes four mathematical domains (arithmetic, order, set membership, propositional logic) through dedicated engines converging in a final logic module. Across 5 seeds, THEIA passes 12/12 Unknown-involving K3 rules at $> 99\%$ per-rule accuracy (39/39 under the full K3 truth table, App. D). K3 learnability itself is not the central finding—a tuned Transformer baseline also passes all 39 rules (App. H); the contributions are instead a *reliability spectrum* under discretized end-to-end composition and an *uncertainty–verdict representation profile*, both established on a controlled synthetic substrate. A sequential composition experiment on an absorbing-state-free mod-3 substrate (deliberately distinct from K3 chains, whose absorbing elements trivialize length generalization; §4.5) generalizes from 5-step training to 500-step evaluation at $99.96\% \pm 0.04\%$ across 5 THEIA seeds. Replacing the four-engine backbone with flat MLPs collapses to chance under the identical discretized training protocol; a larger-capacity ResMLP grid reaches $\geq 99\%$ on only $3/20$ trials, and a larger-capacity 3.58M Transformer reaches $99.24\% \pm 0.34\%$—a four-tier reliability spectrum in which THEIA's cross-seed std is $9\times$–$479\times$ tighter than every comparator (App. G).

Mechanistic probing reveals *asymmetric propagation of uncertainty and verdict*: the network preserves the Has-Unknown signal at every upstream engine boundary, while final-verdict decodability remains at or below a 73.4% U-vs-non-U oracle reference until the Logic boundary (linear and MLP probes). Since both signals traverse the same engines and bridges, the asymmetry reflects a learned representational organization rather than architectural data flow alone. Activation patching further rules out residual-shortcut explanations on matched non-absorbent configurations (§4.3). The Transformer reaches equivalent correctness through a qualitatively different representational trajectory, indicating that modular and attention-based architectures realize distinct compositional strategies.

1. Introduction

Compositional reasoning—combining sub-results from independent domains into a coherent conclusion—is a hallmark of systematic generalization. Symbolic solvers such as Z3 [1] achieve formal correctness over user-specified formulas, but require explicit encodings and do not learn task-specific Unknown-propagation behavior from data. Neuro-symbolic systems (DeepProbLog [2], NeurASP [3], Scallop [4]) bridge the two paradigms, but the actual three-valued or probabilistic inference is delegated to an external symbolic component (a probabilistic logic solver, ASP layer, or Datalog engine) rather than learned as the K3 truth table inside a neural module. Three-valued logic with formal Unknown handling arises naturally in SQL NULL semantics, diagnosis under missing tests, and legal reasoning under indeterminate facts—settings where correctly propagating "I don't know," and knowing when a definite value overrides it, is essential for safe decision-making.

We ask two questions: *can a pure neural network learn complete Kleene three-valued logic—including the non-trivial absorption rules where a definite value overrides an Unknown—without any external symbolic inference engine?* And if so, *what architectural prior enables the resulting compositional computation to generalize to chains 100× longer than training?* These sit at the intersection of compositional learning and interpretability.

[1]Irvine Valley College, Irvine, CA, USA. Correspondence to: Augustus Haoyang Li <augustus@gus.li>.

Accepted to the 2nd Workshop on Compositional Learning at ICML 2026, Seoul, South Korea. Copyright 2026 by the author(s).

We present THEIA[1], a modular neural reasoning engine operating in 128-dimensional vector space. THEIA processes arithmetic, order, sets, and propositional logic through dedicated neural engines. Inputs may be marked Unknown with probability $P = 0.15$; K3 serves not as an application benchmark but as a fully enumerable compositional substrate where correctness, uncertainty propagation, and internal commitment points can be verified exactly. All inference happens inside the network; there is no external solver.

K3 learnability is not unique to THEIA: under appropriate training, a Transformer baseline also learns the full 39-rule K3 truth table at $> 99\%$ per-rule accuracy (App. H), and concurrent differentiable ternary logic-gate networks [29] learn K3 behavior through gate-structured neurons (§2). The central claims are instead the *representation profile* (uncertainty–verdict asymmetric propagation) and the *reliability spectrum* under discretized end-to-end composition; wall-clock convergence is auxiliary. A complete module-level specification sufficient for re-implementation is given in Appendix B.

Contributions.

1. **Architectural inductive biases form a reliability spectrum for compositional generalization.** A mod-3 chain experiment with Gumbel-softmax discretization generalizes from 5-step training to 500-step evaluation at $99.96\% \pm 0.04\%$ across 5 THEIA seeds (§4.5). The mod-3 substrate is deliberately *not* K3—K3's absorbing elements trivialize chain evaluation—so this is evidence about the *architectural prior under discretized composition*, not about K3 chain semantics (§4.5, "Relation to K3 learning"). Controlled ablations (App. G) establish a four-tier reliability spectrum under the identical protocol: flat MLPs (0.80–2.75M) collapse to chance despite matching Phase 1 accuracy; a larger-capacity ResMLP 2×2 grid reaches $\geq 99\%$ on only $3/20$ trials; a 3.58M pre-LN Transformer reaches $99.24\% \pm 0.34\%$; THEIA's cross-seed std is $9\times$–$479\times$ tighter than every comparator, all of which carry *more* capacity, making the advantage a lower bound.
2. **Uncertainty–verdict asymmetric propagation with causal localization.** The network preserves Has-Unknown at every upstream boundary at its closed-form Bayes ceiling, while verdict decodability stays at or below the 73.4% U-vs-non-U oracle reference under both linear and MLP probes (App. E). Both signals traverse the same engines and bridges, so the asymmetry is not forced by data flow alone. Activation patching of $\mathbf{v}_{\text{ord}}$ flips $T \to U$ on $4{,}898/4{,}898$ non-absorbent OR pairs across 5 seeds ($4{,}719/4{,}719$ on AND), ruling out residual shortcuts (§4.3, App. F).
3. **Auxiliary: wall-clock convergence under tested optimizer regimes.** Under per-architecture development defaults (Correction in App. H), THEIA reaches 12/12 Kleene per-rule accuracy in 7.93 ± 1.40 min vs 51.5 ± 11.0 min for a larger, parameter-comparable 8L Transformer (3.64M; 7/8 seeds convergent)—a $6.5\times$ reference gap; the recipe-applied-to-both control gives $4.93\times$, the Transformer's preferred recipe $\sim 3.6\times$. Reported as auxiliary, not a principal contribution.

Falsifiable hypotheses. Each contribution is paired with a concrete falsification criterion (H1–H3, Appendix A).

2. Related Work

Neuro-Symbolic Systems. DeepProbLog [2], NeurASP [3], and Scallop [4] extend logic programming, ASP, and Datalog respectively with neural predicates, but delegate three-valued or probabilistic inference to an external symbolic component. THEIA performs all reasoning, including Unknown propagation and short-circuit absorption, neurally. Neural module networks [16] pioneered modular neural architectures for compositional reasoning. Differentiable alternatives—Logic Tensor Networks [23, 24], Neural Logic Machines [26], and semantic-loss methods [25]—keep inference in-network but operate under two-valued or fuzzy semantics rather than K3. Marra et al. [22] survey the broader neurosymbolic landscape.

Three-Valued Logic in AI. Kleene's strong three-valued logic (K3) [5] underpins SQL's NULL handling and logic programming semantics [10]. Under the stricter generic-MLP criterion used here—no per-neuron K3 gate identity and no external symbolic inference component—we are unaware of prior work demonstrating neural learning of the complete Kleene truth table including short-circuit absorption in both operand orders, from task data under random initialization (concurrent differentiable ternary logic-gate work meeting a different criterion is discussed below). This framing is descriptive rather than a priority claim, and is distinct from in-context demonstration of K3 rules by pre-trained LLMs, which inherit logical priors from natural-language training and which we do not consider "learning" in the same sense.

Scope of comparison. In each reviewed line of work, prior systems either hand-encode K3 gate semantics, delegate K3 inference to an external symbolic component, or operate outside strong Kleene semantics. No K3 gate, short-circuit, or absorption rule is baked into THEIA's initialization; the 39 truth-table entries are learned from task data. The prototype classification head [17] follows stan-

[1]Three-valued Hybrid Engine for Inference Architecture.

dard metric-learning practice and is not a K3-specific bias. Our "pure neural" framing refers to the absence of runtime symbolic delegation and of K3-specific hand-encoded primitives, not to the absence of all architectural priors. The Phase 3 Gumbel-softmax straight-through discretization is a differentiable hard-argmax over THEIA's own output distribution—not an external symbolic engine, and not hand-encoded K3 rules at runtime. We acknowledge possible gaps in adjacent communities (multi-valued circuit synthesis, fuzzy-logic neural networks, paraconsistent-logic learning).

Three-Valued Neural Networks. Earlier work [18, 19, 20] explored three-valued neural networks from a circuit-synthesis perspective with hand-designed neurons implementing three-valued AND/OR gates and hardcoded multi-valued gate semantics. More recently, Logical Neural Networks [21] hardcode each neuron as a logical connective and apply symbolic Upward–Downward inference at runtime. THEIA differs from all of these: its neurons are generic MLP units, and the Kleene three-valued algebra emerges as a learned representational property rather than architecturally encoded primitives. LNN, the closest in design intent, cannot serve as an empirical baseline: it requires hand-specified formula structure and does not handle numerical functions or equality, so it cannot evaluate $a + b$, $c > d$, or $c \in S$.

Differentiable logic gate networks. Concurrent work on differentiable ternary logic-gate networks [29] extends differentiable logic gate networks [27, 28] to Kleene K3 by parameterizing each neuron over ternary gate families and selecting a gate per neuron post-training. This shares THEIA's goal of neural K3 learning with an Unknown state, but differs in the architectural prior: DTLGN commits each neuron to a K3 gate post-training, whereas THEIA uses generic MLP neurons and learns K3 behavior at the representation/output level. Thus both avoid runtime symbolic delegation, but differ in whether K3 primitives are built into the neuron-level hypothesis class. Conceptually, DTLGN asks whether networks composed of K3-committed gates can learn useful ternary computations; THEIA asks whether K3 behavior can emerge from generic MLP units without committing individual neurons to K3 primitives.

Neural Algorithmic Reasoning and Probing. The CLRS benchmark [6] and Discrete Neural Algorithmic Reasoning (DNAR) [7] target deterministic two-valued algorithmic reasoning; THEIA targets three-valued logic with native Unknown propagation as a learned representational property. The multi-hop and modular-arithmetic results of §4.4 additionally touch on GNN expressivity [8, 9] and over-smoothing [11, 12]; our uncertainty–verdict propagation analysis (§4.3) extends linear-probing methodology [14] to modular reasoning architectures.

3. Architecture

3.1. Overview

THEIA processes a four-domain reasoning chain in a fixed sequential-parallel topology (Figure 1). *Domain-separated encoding* means each domain (arithmetic, order, set, logic) is processed by a dedicated engine with no shared parameters and no cross-domain attention; information flows between domains only through explicit boundary connections: bridge layers and residual bypasses. The forward pass:

$$\mathbf{c} = \text{ArithEngine}(a, b, \oplus) \tag{1}$$

$$\mathbf{v}_{\text{ord}} = \text{OrderEngine}(\text{Bridge}_{ao}(\mathbf{c}) + \mathbf{c},\ d,\ R) \tag{2}$$

$$\mathbf{v}_{\text{set}} = \text{SetEngine}(\text{Bridge}_{as}(\mathbf{c}) + \mathbf{c},\ S) \tag{3}$$

$$\mathbf{o} = \text{OutHead}(\text{LogicEngine}(\mathbf{v}_{\text{ord}},\ \mathbf{v}_{\text{set}},\ \odot)) \tag{4}$$

where Bridge_{ao}, Bridge_{as} are residual MLPs (33K params total). Each input is Unknown with probability $P_{\text{unk}} = 0.15$, receiving a learnable embedding (§3.2).

3.2. Domain Engines

All modules operate at D=128. We use a shared two-layer template $\text{MLP}(i{\to}o) := \text{Linear}(i, 2i) \to \text{GELU} \to \text{LayerNorm}(2i) \to \text{Linear}(2i, o)$ and a subspace template $\text{Sub}(i) := \text{Linear}(i, 2D) \to \text{GELU} \to \text{LayerNorm}(2D) \to \text{Linear}(2D, D)$.

Unknown input representation. Each scalar input (a, b, d) is encoded by a numeric encoder $\text{Linear}(1, D/2) \to \text{GELU} \to \text{Linear}(D/2, D)$; the set S by $\text{Linear}(21, D/2) \to \text{GELU} \to \text{Linear}(D/2, D)$. Each encoder owns one learnable vector $\mathbf{u} \in \mathbb{R}^{128}$ (Gaussian init): when an input's Unknown flag is set, the encoder output is *hard-replaced* by $\mathbf{u}$ ($\hat{\mathbf{x}} = \mathbf{u}$ if flagged, else $\text{enc}(x)$). No flag bit is concatenated or routed separately; downstream modules, bridges included, receive Unknown-ness only through these learned 128-dim vectors. This is what makes the upstream Has-Unknown decodability of §4.3 a property of learned features rather than of an explicit flag channel.

Engines. **Arithmetic** (404,736 params): $\text{MLP}(3D{\to}D)$ on $[\text{enc}(a); \text{enc}(b); \text{Emb}_{\oplus}]$ with a 4-entry operator embedding; a, b share one numeric encoder. **Order** (1,179,648 params): on $\mathbf{x} = [\tilde{\mathbf{c}}; \text{enc}(d); \text{Emb}_R]$ ($\tilde{\mathbf{c}}$ the bridged arithmetic vector, R a 6-entry relation embedding), three parallel $\text{Sub}(3D)$ branches with cross-fusion gates (two two-way gates and one three-way gate, each Linear $\to$ GELU $\to$ LayerNorm; exact wiring in Appendix B), then $\text{MLP}(4D{\to}D)$ on the fused branches concatenated with $\tilde{\mathbf{c}}$ (an internal residual). **Set** (208,128 params): $\text{MLP}(2D{\to}D)$

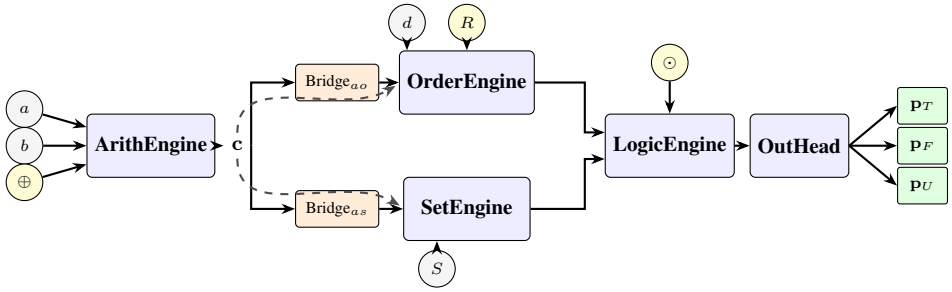

Figure 1. THEIA architecture. ARITHENGINE produces $\mathbf{c}$ from $(a, b, \oplus)$; residual bridges (orange) route $\mathbf{c}$ to ORDERENGINE (d, R) and SETENGINE (S), which run in parallel and feed LOGICENGINE under $\odot$; OUTHEAD matches against learned prototypes $\{\mathbf{p}_T, \mathbf{p}_F, \mathbf{p}_U\}$ (orthogonal init, allowed to drift; §3). No cross-domain attention or shared parameters; dashed arrows denote residual bypass.

on $[\tilde{\mathbf{c}}; \mathrm{enc}(S)]$. **Logic** (908,032 params): same three-branch cross-fusion topology as Order on $[\mathbf{v}_{\text{ord}}; \mathbf{v}_{\text{set}}; \mathrm{Emb}_{\odot}]$ (5-entry operator embedding), then MLP($3D \to D$). Exact dimensions, fusion wiring, and a parameter ledger reconciling to the 2,751,232 total are in Appendix B.

On the subspace labels. The development mnemonics (Global/Local/Event) and (Conjunctive/Disjunctive/Implicative) carry *no enforced semantics*—no loss term, routing rule, or initialization ties any branch to its name—and we claim no theoretical motivation for them; the names are retained only for correspondence with the released code. The decomposition is a convergence aid, not a correctness requirement: §4.4 shows a single-MLP Logic Engine also passes 12/12 Kleene rules.

3.3. Output and Multi-Hop Extension

Three prototype vectors $\{\mathbf{p}_T, \mathbf{p}_F, \mathbf{p}_U\} \in \mathbb{R}^{128}$ serve as classification anchors in the style of prototypical networks [17], initialized to be mutually orthogonal (an engineering choice, not from [17]). Classification uses a prototype-matching head with logits $\mathbf{o} \cdot \mathbf{p}_v$, trained with cross-entropy and Unknown upweighting ($w_U = 2.0$). The prototypes are learned `nn.Parameters` and drift from orthonormal during training; the dot-product logits and the $\arg\max_v \cos(\mathbf{o}, \mathbf{p}_v)$ form coincide exactly only in the orthonormal limit, with top-1 agreement tight in the high-margin convergence regime but not separately quantified here. For structural generalization experiments, a bidirectional message-passing GNN replaces the single-step Order Engine, with depth decoupled from chain length.

4. Experiments

4.1. Setup

All experiments use an NVIDIA RTX 5080 (16GB), the AdamW optimizer [15] with cosine annealing and mixed precision (FP16); THEIA trains at lr 10^{-3}, batch 4096 throughout. Four-domain experiments use 2M samples (80/20 train-test split; 400K test samples) with NUM_RANGE $= 20$ and $P_{\text{unk}} = 0.15$. Inputs are integers; arithmetic operands $a, b \sim \mathcal{U}\{1, \ldots, 20\}$, $d \sim \mathcal{U}\{0, \ldots, 20\}$. The arithmetic operator $\oplus$ is uniform over four operations with range-preserving semantics ($c = \min(a+b, 20)$; $c = |a-b|$; $c = \min(ab, 20)$; $c = a \bmod b$), the relation R is uniform over $\{>, <, =, \geq, \leq, \neq\}$, S is a uniformly random subset of $\{0, \ldots, 20\}$ (21-bit mask), and the logic operator $\odot$ is uniform over $\{\wedge, \vee, \rightarrow, \leftrightarrow, \neg\}$. Labels follow strong Kleene semantics with Boolean-OR Unknown propagation: c is Unknown iff a or b is flagged; val_{ord} is Unknown iff c or d is; val_{set} is Unknown iff c or S is; the final verdict applies the K3 truth table of $\odot$. The resulting label distribution is approximately 26% False, 33% True, and 41% Unknown. The Unknown class weight $w_U = 2.0$ was fixed early in development to prioritize fidelity on the K3-critical Unknown class and its low-frequency absorption configurations, and is held constant across every architecture, seed, and ablation in this paper (including all baselines of App. G); we did not sweep it, and its sensitivity is untested (§4.6). Kleene diagnostic tests use 10,000 independently generated samples per rule. Diagnostic construction requires care; the protocol and the two pitfalls we encountered (cross-domain Unknown coupling; relation edge cases) are documented in Appendix C. Multi-hop experiments use 1M samples with 5-hop training chains. All multi-seed experiments use seeds $\{42, 123, 256, 777, 999\}$; Transformer baseline experiments are extended to 8 seeds with $\{31415, 27182, 14142\}$ where noted.

Transformer baseline configurations. We use two 8-layer, 8-head, d=192 Transformer variants (3.58–3.64M parameters): a post-LN configuration (BigTransformer) for the Kleene-task and probing comparisons (§4.2, §4.3), and a pre-LN variant (TF8LTuned) for the chain pipeline (§4.5). The BigTransformer trains under its lineage development configuration (AdamW lr 5×10^{-4}, batch 2048; Correction in App. H); in the chain pipeline it fails to converge at the shared rate 10^{-3} (App. G).

4.2. End-to-End Three-Valued Algebraic Learning

Under a shared early-stopping criterion (overall accuracy $> 99.9\%$ AND all 12 Kleene rules $> 99\%$ on two consecutive checkpoints), the 8L Transformer reaches 12/12 on 7 of 8 attempted seeds (§4.1) with mean wall-clock 51.5 ± 11.0 min over the converged seeds. The remaining seed (123) fails within the 150-epoch budget but converges under Transformer-specific tuning (Appendix H)—hyperparameter sensitivity, not an architectural limitation. THEIA reaches 12/12 on 5/5 seeds in 7.93 ± 1.40 min—a $6.5\times$ wall-clock gap across development defaults. The baseline carries $\sim 32\%$ more parameters (3.64M vs 2.75M), so THEIA's efficiency is not explained by parameter count.

Why Kleene accuracy is the primary metric. Kleene short-circuit and absorption rules occur at low natural frequency ($F \vee F < 0.8\%$), so bulk overall accuracy can be satisfied while edge-case rules remain unreliable. We therefore use per-rule Kleene diagnostic as the primary metric and a Kleene-aware stopping criterion throughout.

Optimizer regimes (corrected; App. H). Each architecture ran its development default (§4.1), so the $6.5\times$ figure is a development-defaults reference point, not an optimizer-controlled comparison. Rerunning the Transformer with a Transformer-standard recipe (peak lr 10^{-4}, β_2=0.98, linear warmup) gives $\sim 3.6\times$ on the Kleene-aware milestone. The controlled anchor is the recipe-applied-to-both setting: $4.93\times$ (Welch CI $[4.40, 5.66]$); a symmetric tuned-vs-tuned comparison remains future work (App. H).

Targeted diagnostic verification. We construct samples with precisely controlled inputs to verify each Kleene rule independently (10K samples/rule; a rule *passes* above 99%). All 12 Unknown-involving rules pass across all 5 seeds (60/60), with commuted absorption rules verified in both operand orders (means 99.93%). Extending the diagnostic to the **full 39-rule Kleene K3 truth table** (App. D) yields **195/195 rule–seed combinations passing at** $> 99\%$ (grand mean 99.88%), supporting the "complete Kleene" claim at the full truth-table level. The tuned BigTransformer baseline likewise passes all 39 rules at $> 99\%$ across three seeds (117/117, App. H), providing a matched full-coverage comparison.

4.3. Uncertainty–Verdict Asymmetric Propagation: Probe Evidence and Causal Verification

We extract hidden representations at *domain boundaries*—the output of each engine after its full internal computation—and analyze them using linear SVM probes and inter-class Euclidean distance (Table 2). All probing analyses use the same 5 checkpoints as Table 5, 50K samples per checkpoint; results are mean $\pm$ std across seeds. Since the logic operator enters only at the Logic Engine by construction (Eqs. 1–4), upstream verdict probes are partially architecturally bounded; the Has-Unknown signal, by contrast, can reach every upstream boundary through the input flags on a, b, d. The non-trivial finding is a bounded verdict side paired with a ceiling-level uncertainty side along the same pathway.

Verdict-side vs. uncertainty-side decodability. At the Arithmetic boundary, linear-probe accuracy for the 3-class verdict is 60.9%, below the 73.4% accuracy achievable by an oracle that perfectly separates Unknown from non-Unknown samples (the *U-vs-non-U oracle reference*; derivation from the empirical VU joint distribution in App. E). This reference is strictly weaker than the Has-Unknown input flag, since Kleene absorption can make the verdict definite when inputs carry Unknown. The Order and Set boundaries reach 67.2% and 69.7%, still below this reference; accuracy then jumps to 99.9% at the Logic boundary. To rule out linearly-inaccessible verdict information, we train a 2-hidden-layer MLP probe (App. E); the nonlinear probe reaches 61.1% / 67.6% / 70.4% / 100.0% at Arith/Order/Set/Logic—a $+0.7$pp worst-case gain over the linear probe, never crossing the 73.4% reference upstream. By contrast, the Has-Unknown signal is decodable at every upstream boundary well above the $\approx 52\%$ majority baseline: $80.0\%/91.1\%/90.8\%/99.7\%$ at Arith / Order / Set / Logic (std < 0.001). Since the input flags on a, b, d traverse the same engines that the verdict would, the asymmetry is not a routing-structure consequence but a learned representational organization: the network preserves input-level uncertainty through each engine while deferring verdict commitment to the Logic boundary. The F–T centroid distance grows $1{,}898\times$ from Arith to Logic (Table 2), corroborating this picture; raw Euclidean distances should not be compared across architectures due to differing hidden dimensions and activation scales.

Theoretical ceiling on Has-Unknown decodability. The empirical Has-Unknown accuracies $80.0\%/91.1\%/90.8\%/99.7\%$ saturate the closed-form Bayes ceiling at each boundary ($0.7995/0.9079/0.9079/1.0$ under the 4-flag $\mathrm{Bernoulli}(0.15)$ generator) within sampling/rounding error (≤ 0.3pp) upstream, while the verdict-side bound stays at or below the 73.4% U-vs-non-U oracle reference upstream—consolidating the asymmetry as ceiling-vs-bound rather than two empirical numbers. Full derivation in App. E.

Transformer baseline layer-wise probing. For the Table 1 Transformer baseline, verdict decodability reaches near-perfect accuracy by layer 4 and the F–T centroid distance contracts before expanding—a trajectory qualitatively unlike THEIA's boundary-localized pattern (full layer-wise

Table 1. Four-domain reasoning accuracy, each architecture at its development default (THEIA: 10^{-3}/4096; Transformer: 5×10^{-4}/2048; Correction in App. H). THEIA over 5 seeds; Transformer over 7/8 converged seeds (see †). Both architectures pass all 12 Kleene diagnostic rules at $> 99\%$ on every converged seed. Tuned controls in App. H.

Model	Accuracy	Parameters	Seeds 12/12	Time (min)
THEIA (modular)	$99.96\% \pm 0.01\%$	2.75M	5/5	$\mathbf{7.93 \pm 1.40}$
Transformer (8L, 8H)	$99.98\% \pm 0.01\%$	3.64M	7/8	51.5 ± 11.0 †
THEIA wall-clock ratio, development defaults (same-recipe control: $4.93\times$, App. H)				$\mathbf{6.5}\times$

†7/8 Transformer seeds converged under the development-default setting; the failed seed converges under Transformer-standard tuning (App. H).

Table 2. Verdict decodability and representational separation by domain boundary. Mean $\pm$ std over 5 trained checkpoints (seeds $\{42, 123, 256, 777, 999\}$); 50K samples per checkpoint. SVM accuracy is the linear probe accuracy on a 3-class classification task; F–T distance is the Euclidean distance between class centroids in the 128-dim representation space; separation ratio is the mean of per-seed (Logic/Arithmetic) F–T distance ratios.

Domain Boundary	SVM Accuracy	F–T Distance	Separation Ratio
Arithmetic	0.609 ± 0.001	0.146 ± 0.015	—
Order	0.672 ± 0.002	3.130 ± 0.302	—
Set	0.697 ± 0.002	5.967 ± 0.643	—
Logic	$\mathbf{0.999 \pm 0.000}$	$\mathbf{272.738 \pm 37.070}$	$\mathbf{1{,}898}\times$

numbers and caveats in App. E, Table 9).

Mechanistic probing. To trace information flow, we train linear probes at each domain boundary to decode six intermediate variables (full list and results in Appendix E, Table 7).

Three observations stand out. **(1) Operator routing is architecturally respected**: the logic operator is undecodable before the Logic Engine (20.2% = chance for 5 classes). This is an architectural sanity check (Eqs. 1–4 do not route the operator upstream), confirmed to hold under training—no learned side-channel emerges—rather than an independent finding. **(2) Late-stage numerical collapse**: arithmetic R^2 stays high through Arith and Order ($\sim$0.83), drops at Set (0.81), and collapses at Logic (0.16). **(3) The network learns to preserve uncertainty while verdict remains latent**: Has-Unknown accuracy is $\geq$80.0% at every upstream boundary (vs. $\approx 52\%$ majority baseline), reaching 99.7% at the Logic output, while verdict decodability stays at or below the 73.4% U-vs-non-U oracle reference until the Logic boundary. Because both signals traverse the same engines and bridges, (3) is the non-trivial representational finding: the architecture permits upstream verdict decodability in principle (uncertainty demonstrates the pathway is traversable), but the network commits to the verdict only at the Logic boundary. The subspace structure documented here is specific to THEIA among the tested architectures; Appendix G demonstrates that flat-MLP backbones collapse under the end-to-end discretized training of §4.5, while both THEIA and a Transformer baseline remain stable.

Causal verification via activation patching: ruling out residual shortcuts. The probing results above are descriptive. A natural concern is that the Logic Engine might route decisions through a residual connection from $\mathbf{c}$ or the input flags, making the Order/Set engine outputs informationally redundant. To test this, we construct matched $(T\vee F,\ U\vee F)$ pairs that are identical except for the Order-Engine output, and feed the U-side $\mathbf{v}_{\text{ord}}$ into the T-side forward pass. If $\mathbf{v}_{\text{ord}}$ were functionally redundant under a residual shortcut, the prediction would remain T; in fact, the patched prediction flips from T to U on **4,898/4,898 OR pairs** aggregated across $n = 5$ seeds (per-seed minimum 100.0%), and generalized to AND on matched $(T\wedge T,\ U\wedge T)$ pairs (4,719/4,719). The residual-shortcut hypothesis is therefore rejected on non-absorbent $T \to U$ configurations: under these operand patterns the Logic Engine's output is mediated by its designated Order/Set inputs rather than bypassing them via residual. We do not read this as an independent proof of verdict localization—the experiment is tightly controlled by construction—but as a necessary precondition for the uncertainty–verdict asymmetry of Table 2 to be interpretable as a representational finding. Construction details, the numerical equality verification, and per-seed eligibility numbers are in Appendix F.

Operator-identity decomposition of the Set-boundary lift. On the non-Unknown subset, the Set-boundary verdict probe reaches 60.13% (+4.09pp above NU majority). A three-way decomposition (App. E) shows the lift is neither operator encoding (operator-identity probe at chance) nor per-operator verdict leakage (stratified probes *below*

per-operator majority), but val_{set}'s Kleene-correlation with the verdict conditional on the operator. The persistent upstream signals are uncertainty and each engine's designated variable; the verdict remains Logic-exclusive.

4.4. Ablation Studies

Subspace decomposition: replacing the Logic Engine's three parallel C/D/I subspaces with a single MLP (2.49M total params) still achieves 99.97% overall accuracy and passes 12/12 Kleene rules ($> 99\%$, worst $F \vee U$ at 99.73%); the decomposition aids convergence but is not the source of correctness. **Bridge layers**: removing the cross-domain bridges (33K params, 1.2% of total) yields a 2.72M model also passing 12/12 ($> 99\%$, worst $T \wedge U$ at 99.94%); whether bridges become load-bearing under the discretized end-to-end training of §4.5 is not tested here. **Unknown-probability cross-distribution**: a model trained at P_{unk}=0.05 and evaluated at P_{unk}=0.50 (10× shift) keeps 99.9975% overall accuracy and 12/12 Kleene rules, ruling out fitting to the training Unknown distribution. **Multi-hop chain reasoning**: a GNN extension on transitive order chains achieves 99.99% accuracy from 5 to 50 hops, but 99.6% of chains are decided by the first non-identity edge (decisive depth ≈ 0.5) due to absorbing states; modular arithmetic on graphs (mod 5) requiring true global aggregation could not be learned, motivating the absorbing-state-free experiment of §4.5.

4.5. Sequential Composition Generalization

Although THEIA learns K3 at the rule level (§4.2), K3's absorbing elements ($T \vee X = T$, $F \wedge X = F$) make long Kleene chain evaluation a shortcut: any chain with an absorbing element resolves trivially. We confirm this empirically (§4.4). To isolate *structural* generalization from absorption shortcuts, we ask: can THEIA generalize sequential local composition to arbitrary length when absorbing states are eliminated?

Task design. We replace Kleene chain composition with modular arithmetic: $\text{state}_t = (\text{state}_{t-1} + \text{local_verdict}_t) \bmod 3$. This has no absorbing states, a non-trivial 9-entry transition table, and approximately uniform state distribution at all chain depths.

Architecture. The chain THEIAStep is a re-instantiated 1.51M step computer trained from scratch in the chain pipeline, not initialized from the 2.75M four-domain checkpoint; its smaller size comes from a single-subspace Logic stage and a chain-specific scaled-cosine head (full specification in App. B). A 4,803-param transition MLP is trained on top. Gumbel-softmax straight-through [30, 31, 32] discretizes the state at each step (hard one-hot forward, smooth backward), so step 500 receives an input in the same discrete one-hot form as step 5.

Three-phase training. Phase 1 trains THEIAStep independently to $\geq 99.9\%$ local accuracy on 2M samples (∼50 epochs); a plateau-restart mechanism (accuracy <90% after 40 epochs ⇒ reinitialize) ensures robust convergence. *Plateau-restart disclosure:* 4/5 seeds $\{42, 123, 256, 777\}$ reached the criterion in one try; seed 999 triggered the restart, converged on the second try, and consumed ∼2× Phase 1 compute. The strict 4-seed aggregate excluding seed 999 equals the 5-seed aggregate within reported precision ($99.96\% \pm 0.04\%$). The thresholds were engineering parameters introduced without pre-registration; the same mechanism does not rescue the post-LN BigTransformer's chain non-convergence at the shared rate (Table 1), so it is not an architecture-specific advantage. Threshold-sensitivity sweeps are left to future work. Phase 2 freezes THEIAStep and teacher-forces the 4,803-param transition network to 9/9 mod-3 addition entries (2 epochs). Phase 3 unfreezes all parameters and fine-tunes end-to-end on 5-step chains with Gumbel-softmax (∼30 epochs to 100%). The Gumbel-softmax temperature follows a linear decay schedule $\tau = \max(0.1,\ 0.5 - 0.01\,\text{ep})$: τ decreases from 0.5 at epoch 1 to the floor 0.1 at epoch 40 and is clamped there afterward; the same schedule is used, unchanged, by every backbone in App. G.

Results. Table 3 summarizes generalization performance across 5 seeds.

Table 3. Sequential composition generalization (mod-3, no absorbing states). Trained on 5-step chains, tested on 10–500 steps. Results over $n = 5$ seeds $\{42, 123, 256, 777, 999\}$.

Chain Length	Accuracy (mean ± std)	All ≥99%
5 (in-dist.)	100.00% ± 0.00%	✓
10	100.00% ± 0.00%	✓
50	100.00% ± 0.00%	✓
100	100.00% ± 0.00%	✓
500	**99.96% ± 0.04%**	✓

All 5 seeds lie in the 99.90–99.99% range at 500 steps (per-seed values: $\{99.90, 99.99, 99.97, 99.99, 99.96\}\%$; seed 999 reached this via Phase 1 auto-restart—see plateau-restart disclosure above). The non-trivial requirement is sustaining ≥99.9% local verdict accuracy *under Phase 3 end-to-end Gumbel-softmax training*: THEIA and the Transformer do, the tested flat MLPs do not (App. G). Three properties combine to enable length generalization: (1) high local verdict accuracy (≥99.9%) so each step receives correct input; (2) exact transition function (9/9 entries); (3) Gumbel-softmax discretization preventing error accumulation by snapping intermediate states to clean one-hot codes (without it, $0.999^{500} \approx 60.6\%$ would dominate). Property (3) is largely a mathematical consequence of straight-through dis-

cretization; the architectural contribution lies in sustaining (1) under that same discretized training regime, where flat MLPs fail. The 500-step result is *not* trivially explained by absorbing states: empirical state frequencies confirm approximate uniformity at all measured chain depths (max deviation 0.62 pp at step 100; details in App. G).

Relation to K3 learning. The chain THEIAStep is re-instantiated and trained from scratch; it does not inherit the K3-trained checkpoint, and mod-3 addition is not Kleene composition. The result is therefore *not* evidence that THEIA performs 500-step K3 chain reasoning—no clean test of that claim exists, because K3's absorbing elements resolve 99.6% of chains at the first non-identity edge (§4.4), making long-K3-chain accuracy uninformative about composition. What connects §4.2 and this section is the *architectural prior*, not the logic: §4.2 shows the four-engine prior learns the complete K3 algebra at the rule level; this section shows the same prior, on the minimal absorbing-free 3-state substrate sharing the identical pipeline and training protocol, sustains *reliable learned discrete-state composition* where structure-free backbones with more capacity fail (App. G). These are two properties of one inductive bias, established on two deliberately different substrates; the title's "complete Kleene" claim rests on §4.2 alone, and the length-generalization claim rests on this section alone.

Boundary characterization. Three experiments define a boundary: transitive chains (local + absorbing) generalize trivially; mod-3 composition (local, no absorbing) generalizes to 500 steps; modular arithmetic on graphs (global) cannot be learned. The critical distinction is *computation locality*, not chain length.

4.6. Limitations

All experiments use synthetic benchmarks for precise control and ground-truth verification [6, 7]; adapting THEIA to external benchmarks is future work, the most direct candidate being SQL NULL predicate evaluation, where TRUE/FALSE/UNKNOWN semantics arise natively rather than from a synthetic flag.

Task–architecture alignment. The synthetic task decomposes exactly into arithmetic, order, set, and logic components, and THEIA is given exactly those four engines. The reliability spectrum is therefore evidence for the value of a *matched* modular prior—given a correct decomposition, modularity buys cross-seed reliability that capacity does not (App. G)—and not evidence that THEIA discovers decompositions or that the advantage survives a mismatched or perturbed factorization. Experiments with deliberately misaligned, merged, or permuted engine-domain assignments, and comparisons against alternative modular decompositions [33], are the most important unrun controls.

Other scoping. The Unknown class weight w_U=2.0 is held fixed everywhere but never ablated; per-rule Kleene accuracy on rare absorption configurations may be sensitive to it. The 500-step result is *reliable learned discrete-state composition*: it depends on straight-through discretization (without it, $0.999^{500} \approx 60.6\%$ error compounding would dominate), and a no-discretization ablation is not run—continuous chaining is a different regime, not a controlled variant of this one. The Transformer comparison uses two LN variants (post-LN for §4.2–§4.3, pre-LN for §4.5) because the post-LN variant fails to converge in the chain pipeline at the shared rate (App. G); baseline selection is thus conditioned on trainability under the shared protocol rather than principled per-pipeline tuning, an asymmetry that favors the Transformer where it binds. Likewise, the §4.2 wall-clock comparison is development-defaults, not optimizer-controlled (App. H). The sequential composition experiment uses a minimal 3-class state; validation on richer state spaces (mod-k for $k > 3$, FSA traces, learned codebooks) is deferred. The ResMLP probe spans a 2×2 grid at a single parameter scale; the "residual-only unreliable" reading is scoped to this grid, and the $\geq$99% reliability claim is supported at the aggregate (config, seed) level (scope details in App. G). Global computation (modular arithmetic on graphs) remains out of reach for local message passing, and numerical range generalization fails at $5\times$ the training range [13]: the arithmetic engine learns within-range approximation, and mechanistic claims (§4.3) concern propagation of this bounded-range representation. Activation patching covers non-absorbent configurations only; absorbent cases are vacuous, as the verdict is invariant to the patched operand.

5. Discussion

Architectural inductive biases form a reliability spectrum. The backbone ablation (Appendix G) reveals a gradient rather than a binary. Flat MLPs at both tested capacities match THEIA's local Kleene accuracy within 0.04% yet collapse to chance over 50+ steps under Gumbel-softmax end-to-end training. The ResMLP 2×2 grid partially recovers with a strong depth effect—$\approx$11pp from doubling depth at fixed budget, with expansion nearly neutral—yet only $3/20$ (config, seed) trials cross $\geq 99\%$ ($2/19$ strict; Table 13). The pre-LN Transformer ($99.24\% \pm 0.34\%$) and THEIA ($99.96\% \pm 0.04\%$) both sustain near-perfect length generalization, at different reliability levels. Two axes separate: (i) *mean accuracy*—residual depth recovers from $\sim$33% to $\sim$98% yet reliable $>99\%$ eludes every tested configuration; (ii) *reliability*—THEIA's cross-seed std is $9\times$ tighter than the Transformer's and $36\times$–$479\times$ tighter than individual ResMLP configurations. Because every compara-

tor carries more parameters than THEIA's chain step, the advantage is a lower bound rather than a parameter-matched comparison.

Why might a matched modular prior buy reliability under discretization? (A hypothesis.) The straight-through estimator's backward pass is biased with respect to the discrete forward computation [32, 30], so Phase 3 injects systematically distorted gradients into a backbone whose local function was learned under clean Phase 1 gradients. In a flat MLP every parameter participates in every sub-computation, so this distortion perturbs the entire learned function jointly, and Phase 1 competence is unprotected—consistent with the flat MLPs matching Phase 1 accuracy yet collapsing (Table 12). THEIA's domain-separated parameters confine cross-component interference to the 128-dim engine interfaces: a distorted gradient arriving through the Logic Engine can reshape an upstream engine only through a single bottlenecked boundary vector, not through shared weights. This reading is consistent with evidence that monolithic networks trained end-to-end often fail to organize into functionally modular sub-networks even on decomposable tasks [34], and with the finding that explicit modular architectures help most precisely when the imposed decomposition matches the task's true factorization [33]—the matched regime tested here (§4.6). The ResMLP depth effect fits the same picture as weaker, generic staging—more blocks over which STE-induced drift can be locally corrected, cf. discretized states and deep gate stacks in [7, 27]—but generic depth does not recover the cross-seed stability the matched prior provides. We have not run the isolating experiment (e.g., per-engine gradient-norm tracking during Phase 3); this is the most economical explanation consistent with all four tiers of Table 12, not a demonstrated cause.

Uncertainty–verdict asymmetric propagation as a representational signature. Operator identity is routed only into the Logic Engine and the verdict is a function of the operator, so upstream verdict decodability is architecturally bounded. Against this, the non-trivial finding (§4.3) pairs (a) verdict decodability staying at or below the 73.4% oracle reference at every upstream boundary even under MLP probes biased in their favor, with (b) Has-Unknown decodability saturating the closed-form Bayes ceiling at every boundary (§4.3); activation patching rules out residual shortcuts on the tested non-absorbent configurations. A functional reading: the architecture makes upstream verdict commitment impossible to *complete* (the operator is invisible), but nothing forces uncertainty *preservation*—a network could discard input-flag information at any boundary without violating Eqs. 1–4. That every THEIA boundary instead encodes Has-Unknown at its Bayes ceiling suggests the network learns representations matched to the task's conditional-independence structure: each stage retains exactly the inputs that downstream K3 absorption decisions will consume, and verdict formation coincides with the first point at which its inputs are complete. Probes [14] measure decodability rather than downstream use, which is why the causal half of this claim rests on the activation-patching experiment rather than on probe accuracies alone.

Modular vs. attention-based compositional strategies. The pre-LN TF8LTuned Transformer sustains 99.24% at 500 steps under the identical Gumbel-softmax protocol. THEIA exhibits monotone upstream-to-downstream representational separation (Table 2); the Transformer's token embeddings already exceed the oracle reference, and F–T centroid distance contracts before expanding (Table 9). The two architectures reach comparable correctness through qualitatively different representational trajectories. In systematic-generalization terms [35, 36], both pass a productivity-style test on this substrate—generalizing to compositions far longer than any seen in training—but with different reliability margins, and the deferred-verdict organization of §4.3 is specific to the modular trajectory among the architectures probed. Whether either trajectory survives when the task decomposition is not architecturally given is precisely the open question posed by [33] and deferred to future work in §4.6.

6. Conclusion

We asked whether a pure neural network can learn complete Kleene three-valued logic—no runtime symbolic delegation, no K3-specific hand-encoded primitives—and what architectural prior makes the resulting computation compose reliably. On a fully enumerable synthetic substrate, THEIA learns the complete 39-rule K3 truth table including both operand orders of the absorption rules ($195/195$ rule–seed combinations $> 99\%$); probing plus activation patching reveal a consistent internal organization in which input-level uncertainty is preserved at every engine boundary at its Bayes ceiling while verdict commitment is deferred to, and causally mediated at, the Logic Engine. Under discretized end-to-end composition, the matched modular prior converts into reliability: $99.96\% \pm 0.04\%$ at 500 steps, with cross-seed variance $9\times$–$479\times$ tighter than capacity-favored comparators. K3 learnability itself is shared with a tuned Transformer (App. H) and pursued concurrently by gate-structured ternary networks [29]; THEIA's contribution is the reliability spectrum and the uncertainty–verdict representation profile, scoped to a synthetic, decomposition-matched, discretized regime. The immediate next steps are SQL NULL evaluation (the most direct external test, §4.6), decomposition-mismatch controls, and a symmetric tuned-vs-tuned comparison.

A. Falsifiable Hypotheses (Detailed)

We make three concrete claims, each paired with the evidence that would falsify it.

H1 (structured bias for compositional generalization): replacing the four-engine backbone with a flat MLP (App. G specifies both the 0.80M and 2.75M configurations, the latter $\approx 1.8\times$ larger than THEIA's 1.51M chain step), holding all other pipeline components fixed, collapses Phase 3 Gumbel-softmax end-to-end training to chance. A larger-capacity ResMLP probe across a 2×2 depth×expansion grid (4 configurations at 2.77–2.85M parameters, all larger than THEIA's chain step; $\{4, 8\}$ blocks $\times$ $\{2d, 4d\}$ expansion; 5 seeds each) partially recovers but does not reach reliable $\geq 99\%$ generalization: mean accuracy varies strongly by depth (4-block configs: $86.15\% \pm 9.17\%$ and $87.49\% \pm 19.16\%$; 8-block configs: $97.97\% \pm 1.43\%$ and $97.95\% \pm 2.03\%$), yet only $3/20$ (config, seed) trials reach $\geq 99\%$ as-specified ($2/19$ strict no-restart). *Scope:* this ResMLP probe covers 4 configurations within a 2×2 grid at a single parameter scale; deeper architectures (≥ 12 blocks), wider expansions ($\geq 8d$), pre-vs-post-LN variants, and DenseNet-style multi-scale residuals are not tested. The "residual-only unreliable" reading is scoped to this grid. A Transformer baseline (3.58M) sustains $99.24\% \pm 0.34\%$ (3/5 seeds $\geq 99\%$), and THEIA sustains $99.96\% \pm 0.04\%$ (5/5 seeds in the 99.90–99.99% range). *Falsified by*: a flat-MLP or ResMLP variant sustaining $\geq 99\%$ 500-step chain accuracy on a majority of seeds under the identical discretized training protocol with any Phase 3 learning-rate or depth/width/expansion setting beyond the tested grid (App. G).

H2 (uncertainty–verdict asymmetric propagation with causal localization): (a) *verdict-side bound*: upstream engine outputs do not encode the True/False distinction in a form decodable by a standard 2-layer MLP probe, beyond the 73.4% U-vs-non-U oracle reference, at any of Arith/Order/Set boundaries; (b) *uncertainty-side preservation*: the Has-Unknown signal is decodable at every upstream boundary at accuracy $\geq 75\%$ (substantially above the $\approx 52\%$ majority baseline), demonstrating that the information pathway through the engines does carry input-level uncertainty; (c) *causal localization*: the Logic Engine's output on non-absorbent $T \to U$ uncertainty-propagation configurations is not mediated by a residual shortcut bypassing the designated engine outputs, as verified by activation patching on matched identical-input pairs. *Architectural vs. learned:* operator-identity undecodability upstream ($\approx 20\%$ chance) is fixed by construction (Eqs. 1–4) and functions as a sanity check, not an independent prediction of H2; likewise, the 73.4% verdict bound is partially determined by data flow, so the non-trivial content of (a) is paired with (b), not stand-alone. *Falsified by*: either an MLP probe exceeding 73.4% verdict accuracy at the Arithmetic, Order, or Set boundary (§4.3, Appendix E); or Has-Unknown probe accuracy falling below 75% at any upstream boundary (Appendix E, Table 7); or activation patching of $\mathbf{v}_{\text{ord}}$ between matched $T \vee F$ and $U \vee F$ pairs failing to flip the prediction on a non-trivial fraction of pairs (§4.3, Appendix F).

H3 (development-defaults convergence speed): under each architecture's development-default configuration (THEIA: AdamW lr=10^{-3}, cosine, batch 4096, 200-epoch budget; Transformer: AdamW lr=5×10^{-4}, cosine, batch 2048, 150-epoch budget; neither tuned for this task—see the Correction in App. H), THEIA reaches 12/12 Kleene coverage at materially lower wall-clock cost than a larger, parameter-comparable 8L Transformer (3.64M params, $\sim$32% more parameters than THEIA) on the same seed set. *Scope:* H3 is bounded under three optimizer regimes (App. H). In the development-defaults setting the gap is $6.5\times$. In the Transformer-tuned setting, the Transformer uses its preferred recipe while THEIA stays at its development default, narrowing the gap to $\sim 3.6\times$. In the recipe-applied-to-both control—the only regime in which both architectures share a recipe—the same Transformer-standard recipe is also applied to THEIA (which slows it by $\sim 25\%$ relative to its own default), yielding $4.93\times$ (Welch, 95% CI $[4.40, 5.66]$). A symmetric tuned-vs-tuned comparison (each architecture at its own optimum) is not performed and is left to future work; these regimes bound the comparison rather than establish a fully symmetric result. *Falsified by*: a Transformer configuration matching THEIA's 7.93 ± 1.40 min Kleene-aware convergence on a majority of seeds under a recipe shared by both architectures (§4.2, App. H). Extended to $n = 8$ seeds (7/8 converge, 51.5 ± 11.0 min), the $6.5\times$ development-defaults gap persists; the Transformer-only-tuned follow-up ($n = 3$) narrows it to $\sim 3.6\times$ on the Kleene-aware criterion but does not close it.

B. Complete Architecture Specification

This appendix specifies THEIA at the module level, sufficient to re-implement the architecture from the paper alone; it consolidates and extends §3.2. All boundary representations are D=128-dimensional. Two templates recur: MLP($i \to o$) denotes Linear$(i, 2i) \to$ GELU $\to$ LayerNorm$(2i) \to$ Linear$(2i, o)$, and Sub$(3D)$ denotes the branch template Linear$(3D, 2D) \to$ GELU $\to$ LayerNorm$(2D) \to$ Linear$(2D, D)$.

Input encoders and the Unknown representation. Each numeric encoder is Linear$(1, D/2) \to$ GELU $\to$ Linear$(D/2, D)$ applied to the scalar input scaled by 1/NUM_RANGE (i.e., $x/20$); the set encoder is Linear$(21, D/2) \to$ GELU $\to$ Linear$(D/2, D)$ applied to the 21-bit membership mask. Every encoder instance owns

a learnable vector $\mathbf{u} \in \mathbb{R}^{128}$ (initialized i.i.d. standard normal). When an input's Unknown flag is set, the encoder output is *hard-replaced* by $\mathbf{u}$ (an elementwise `torch.where` on the flag): downstream modules receive only the 128-dim vector, with no separate flag side-channel, so all Unknown-propagation behavior must be carried by the learned geometry of $\mathbf{u}$ relative to the value encodings. The Arithmetic Engine shares one numeric encoder (and one $\mathbf{u}$) between a and b; the Order Engine owns a separate numeric encoder (and $\mathbf{u}$) for d; the Set Engine owns the set encoder (and its $\mathbf{u}$). No encoder or engine parameters are shared across engines.

Arithmetic Engine (404,736 params). $\mathbf{c} = \mathrm{MLP}(3D{\to}D)\big([\mathrm{enc}(a);\ \mathrm{enc}(b);\ \mathrm{Emb}_{\oplus}]\big)$, where $\mathrm{Emb}_{\oplus}$ is a 4-entry $\times$ 128-dim operator embedding.

Three-branch cross-fusion template (Order and Logic Engines). On input $\mathbf{x} \in \mathbb{R}^{3D}$, three parallel branches $\mathbf{h}_k = \mathrm{Sub}_k(\mathbf{x})$, $k \in \{1,2,3\}$, are fused through two two-way gates $\mathrm{Fuse}_j^{(2)} : \mathrm{Linear}(2D, D) \to \mathrm{GELU} \to \mathrm{LayerNorm}(D)$ and one three-way gate $\mathrm{Fuse}^{(3)}$: $\mathrm{Linear}(3D, D) \to \mathrm{GELU} \to \mathrm{LayerNorm}(D)$:

$$\mathbf{h}'_1 = \mathrm{Fuse}_1^{(2)}([\mathbf{h}_1; \mathbf{h}_3]), \qquad \mathbf{h}'_2 = \mathrm{Fuse}_2^{(2)}([\mathbf{h}_2; \mathbf{h}_3]),$$
$$\mathbf{h}'_3 = \mathrm{Fuse}^{(3)}([\mathbf{h}_3; \mathbf{h}'_1; \mathbf{h}'_2]).$$

The third branch thus acts as a hub: it conditions the refinement of the first two and is then refined by their fused versions. The development mnemonics Global/Local/Event (Order) and Conjunctive/Disjunctive/Implicative (Logic) name $(\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3)$ in this template and carry no enforced semantics (§3.2).

Order Engine (1,179,648 params). Input $\mathbf{x} = [\tilde{\mathbf{c}};\ \mathrm{enc}(d);\ \mathrm{Emb}_R]$, where $\tilde{\mathbf{c}} = \mathrm{Bridge}_{ao}(\mathbf{c}) + \mathbf{c}$ and Emb_R is a 6-entry relation embedding. After the cross-fusion template, $\mathbf{v}_{\mathrm{ord}} = \mathrm{MLP}(4D{\to}D)\big([\mathbf{h}'_1; \mathbf{h}'_2; \mathbf{h}'_3; \tilde{\mathbf{c}}]\big)$; the concatenated $\tilde{\mathbf{c}}$ is an internal residual that re-injects the bridged arithmetic vector at the output stage.

Set Engine (208,128 params). $\mathbf{v}_{\mathrm{set}} = \mathrm{MLP}(2D{\to}D)\big([\tilde{\mathbf{c}}_s;\ \mathrm{enc}(S)]\big)$ with $\tilde{\mathbf{c}}_s = \mathrm{Bridge}_{as}(\mathbf{c}) + \mathbf{c}$. No subspace decomposition.

Logic Engine (908,032 params). Input $\mathbf{x} = [\mathbf{v}_{\mathrm{ord}};\ \mathbf{v}_{\mathrm{set}};\ \mathrm{Emb}_{\odot}]$, where $\mathrm{Emb}_{\odot}$ is a 5-entry operator embedding $(\wedge, \vee, \to, \leftrightarrow, \neg)$. Same cross-fusion template as Order; output $\mathrm{MLP}(3D{\to}D)\big([\mathbf{h}'_1; \mathbf{h}'_2; \mathbf{h}'_3]\big)$, with no residual concatenation. The unary $\neg$ is one of the 5 embedding entries; its label semantics act on $\mathrm{val}_{\mathrm{ord}}$, and the second operand does not affect the target.

Bridges (33,536 params) and output head (16,768 params, plus the 384-param prototype matrix). Each bridge is $\mathrm{Linear}(D, D) \to \mathrm{GELU} \to \mathrm{LayerNorm}(D)$, applied with the additive residual of Eqs. 2–3. The output head is $\mathrm{Linear}(D, D) \to \mathrm{GELU} \to \mathrm{Dropout}(0.1) \to \mathrm{LayerNorm}(D)$, followed by the prototype matrix $\{\mathbf{p}_T, \mathbf{p}_F, \mathbf{p}_U\}$ stored as a 3-entry embedding with orthogonal initialization. Training uses dot-product logits $\mathbf{o}{\cdot}\mathbf{p}_v$ with class-weighted cross-entropy ($w_U{=}2.0$); inference classifies by $\arg\max_v \cos(\mathbf{o}, \mathbf{p}_v)$ (the two coincide in the orthonormal limit; see the drift caveat in §3). Table 4 reconciles these per-module counts to the instantiated model's exact total.

Table 4. Parameter ledger for the four-domain THEIA model. Each module count includes its encoders, embeddings, and Unknown vector(s). The total matches the instantiated model exactly.

Module	Parameters
Arithmetic Engine (a/b enc., $\mathrm{Emb}_{\oplus}$, $\mathbf{u}$)	404,736
Order Engine (d enc., Emb_R, $\mathbf{u}$)	1,179,648
Set Engine (set enc., $\mathbf{u}$)	208,128
Logic Engine ($\mathrm{Emb}_{\odot}$)	908,032
Bridges ($\times$2)	33,536
Output head	16,768
Prototypes $\{\mathbf{p}_T, \mathbf{p}_F, \mathbf{p}_U\}$	384
Total	**2,751,232**

Chain-pipeline THEIAStep (1,508,096 params). The chain step computer of §4.5 differs from the four-domain model in three ways: (i) the Logic stage uses a single MLP rather than the three-branch cross-fusion template; (ii) its numeric inputs enter the encoders unscaled (no $1/\mathrm{NUM_RANGE}$ normalization), unlike the four-domain model above; and (iii) the classification head is Linear $\to$ GELU $\to$ Linear followed by L_2 normalization and scaled cosine similarity (temperature 10) against an axis-aligned prototype matrix, a substitution made because the LayerNorm-then-cosine head exhibited less stable gradient flow under straight-through discretization in preliminary chain experiments (§4.5). The 4,803-parameter transition network is a 3-layer MLP over the concatenated previous state and current local verdict. Exact layer dimensions for the chain variant accompany the released code (App. I).

C. Constructing Valid Kleene Diagnostic Tests

Targeted diagnostic tests must inject the Unknown value into one operand of the final logical connective ($\mathrm{val}_{\mathrm{ord}}$ or $\mathrm{val}_{\mathrm{set}}$) while keeping the other operand definite. During the development of this work we encountered two construction pitfalls that silently corrupted earlier diagnostic runs. We document them here so that future work using similar protocols can avoid the same mistakes; both fixes are applied throughout the experiments reported in this paper.

Pitfall 1: Cross-domain Unknown contamination. To make $\text{val}_{\text{ord}} = \text{Unknown}$, a naive construction sets the first arithmetic operand a to Unknown. However, in our task definition the arithmetic engine produces $c = \text{arith}(a, b)$, and the natural Unknown propagation rule—Boolean OR over the input unknown-flags, $c_{\text{unknown}} = a_{\text{unknown}} \text{ OR } b_{\text{unknown}}$—then forces c to be Unknown as well. Because the Set Engine takes c as input, set_op_unknown becomes True, collapsing val_{set} to Unknown regardless of the constructed set bits. The diagnostic then unintentionally tests $(\text{Unknown}, \text{Unknown})$ instead of the intended $(\text{Unknown}, \text{definite})$. The model's response to an all-Unknown input is governed by an entirely different Kleene rule than the one the experimenter believes they are testing, and the resulting accuracy number is uninterpretable.

Fix. Inject Unknown via d (the comparison operand of the Order Engine) rather than via a or b. Setting $d_{\text{unknown}} = \text{True}$ makes $\text{val}_{\text{ord}} = \text{Unknown}$ without polluting c, so val_{set} remains controllable.

Pitfall 2: Edge-case relation construction. To force $\text{val}_{\text{ord}} = \text{True}$ under the strict-greater-than relation (REL_GT, $>$), one might set $d = \max(0, c - 1)$. This works whenever $c \geq 1$, but fails when $c = 0$: then $d = 0$, and $0 > 0$ evaluates to False. Under the operator semantics of §4.1, $c = 0$ occurs precisely when ($\oplus = \text{SUB}$ and $a = b$) or ($\oplus = \text{MOD}$ and $b \mid a$); ADD and MUL never produce $c = 0$ on $a, b \in \{1, \ldots, 20\}$. The expected fraction under uniform sampling is $\frac{1}{4} \cdot \frac{20}{400} + \frac{1}{4} \cdot \frac{66}{400} = 5.375\%$ (66 of the 400 (a, b) pairs satisfy $b \mid a$, including all 20 a=b pairs), matching the empirically observed $\sim 5\%$. *Correction:* the version of this appendix reviewed at submission misstated the MOD condition as "$a < b$"; for $a < b$, $a \bmod b = a \geq 1$ is nonzero, so that condition never yields $c = 0$. The correct condition is divisibility, $b \mid a$, which covers $a = b$ as well as pairs such as $(6, 3)$. We thank a reviewer for flagging the imprecision; the construction fix below is unaffected. For the affected samples $\text{val}_{\text{ord}} = \text{False}$ instead of the intended True, contaminating any rule whose construction requires $\text{val}_{\text{ord}} = \text{True}$. The contamination is silent: per-rule accuracy looks plausible but is biased downward by exactly the fraction of $c = 0$ samples.

Fix. Use the greater-than-or-equal relation (REL_GTE, $\geq$) with the same $d = \max(0, c - 1)$. This always yields $\text{val}_{\text{ord}} = \text{True}$ since $c \geq \max(0, c - 1)$ for all $c \geq 0$.

Validation. We adopt both fixes throughout this paper. The doubly-fixed diagnostic protocol is what produces the per-rule accuracies reported in Table 5. We verified consistency by re-running an existing checkpoint under the fixed diagnostic and confirming that the new per-rule numbers fall within the noise band of the multi-seed sweep. We urge future work using similar Kleene diagnostic protocols to inspect their construction for analogous coupling effects between domains and analogous edge cases in relation construction.

D. Complete Kleene Diagnostic (Extended)

This appendix contains the full Kleene K3 diagnostic. Table 5 reports per-rule accuracy on the 12 targeted Unknown-involving short-circuit and absorption rules (the main result referenced throughout §4.2). Table 6 extends the diagnostic to the remaining 27 rules covering all other Kleene K3 truth-table entries (36 binary rules + 3 unary NOT rules total). Construction uses the doubly-fixed protocol of Appendix C throughout.

Table 5. Per-rule Kleene three-valued logic diagnostic (12 targeted Unknown-involving rules). Mean ± std across 5 seeds (10,000 samples per rule per seed); "min" is the worst single seed–rule accuracy. †Unknown is the *first* operand. All 60 rule–seed combinations exceed 99%.

Expression	Expected	Mean ± Std (%)	Min (%)
$F \wedge U$	F	99.97 ± 0.06	99.85
$T \wedge U$	U	99.94 ± 0.04	99.89
$U \wedge F^{\dagger}$	F	99.93 ± 0.06	99.84
$U \wedge T^{\dagger}$	U	99.80 ± 0.11	99.69
$T \vee U$	T	99.93 ± 0.08	99.77
$F \vee U$	U	99.92 ± 0.11	99.71
$U \vee T^{\dagger}$	T	99.93 ± 0.06	99.85
$U \vee F^{\dagger}$	U	99.85 ± 0.10	99.68
$F \rightarrow U$	T	99.95 ± 0.03	99.91
$T \rightarrow U$	U	100.00 ± 0.00	99.99
$T \leftrightarrow U$	U	100.00 ± 0.00	100.00
$F \leftrightarrow U$	U	100.00 ± 0.00	100.00
Grand mean (12/12, 5/5 seeds)		**99.94**	—

Across all 5 trained checkpoints, **all 39 rules pass the $> 99\%$ threshold on every seed** (195/195 rule–seed combinations); grand mean across all 39 rules is 99.88%, and the single worst combination is $F \vee F = F$ at seed 999 with 99.15%, still above the 99% threshold.

The lowest-accuracy entry is $F \vee F = F$ at 99.54%. This is below the worst Unknown-involving rule, which is initially counterintuitive (classical disjunction should be "easier" than Kleene short-circuits). The likely explanation is distributional: under our 4-domain training pipeline, the natural distribution of (F, F) inputs to the final disjunction is sparse compared to the Unknown-involving cases, since both operands must come from independently-evaluated upstream domains that happen to both be False. The targeted diagnostic forces this rare configuration, and the small accuracy gap reflects mild distribution shift between training and diagnostic, not difficulty of the underlying rule. All 5 seeds remain above the 99% threshold.

The 27 new rules span all four binary operators ($\wedge, \vee, \rightarrow$

Table 6. Complete Kleene K3 diagnostic: 27 rules not in Table 5. Mean ± std across 5 seeds; 10,000 samples per rule per seed. All 27 rules pass $> 99\%$ on all 5 seeds (135/135 rule–seed combinations).

Rule	Acc (%)	Rule	Acc (%)	Rule	Acc (%)
Definite–definite (18 rules including NOT)					
$F \wedge F$	100.00 ± 0.01	$F \vee F$	99.54 ± 0.27	$F \to F$	99.74 ± 0.12
$F \wedge T$	99.84 ± 0.05	$F \vee T$	99.92 ± 0.06	$F \to T$	99.98 ± 0.03
$T \wedge F$	99.93 ± 0.08	$T \vee F$	99.96 ± 0.03	$T \to F$	99.86 ± 0.12
$T \wedge T$	99.83 ± 0.09	$T \vee T$	100.00 ± 0.00	$T \to T$	99.86 ± 0.07
$F \leftrightarrow F$	99.74 ± 0.12	$F \leftrightarrow T$	99.66 ± 0.15	$\neg F$	99.72 ± 0.13
$T \leftrightarrow F$	99.88 ± 0.11	$T \leftrightarrow T$	99.81 ± 0.10	$\neg T$	99.96 ± 0.03
Unknown–involving (9 rules not in Table 5)					
$U \wedge U$	100.00 ± 0.00	$U \vee U$	100.00 ± 0.00	$U \to F$	99.89 ± 0.10
$U \to T$	99.93 ± 0.07	$U \to U$	100.00 ± 0.00	$U \leftrightarrow F$	100.00 ± 0.00
$U \leftrightarrow T$	100.00 ± 0.00	$U \leftrightarrow U$	100.00 ± 0.00	$\neg U$	100.00 ± 0.00

, $\leftrightarrow$) plus unary negation, covering every truth-table entry not exercised by Table 5. Together with Table 5, they verify every entry of the complete Kleene K3 truth table, justifying the "complete Kleene K3" framing in the title and abstract.

E. Mechanistic Probing Details

To trace how information flows through THEIA's pipeline, we train linear probes at each domain boundary to decode six intermediate variables: arithmetic result (linear regression, R^2), order truth value, set truth value, final verdict, logic operator identity, and the presence of any Unknown flag in the input (all classification accuracy). Table 7 reports the full per-probe per-boundary results referenced from §4.3.

Bayes-ceiling derivation for Has-Unknown probes. To make the uncertainty-side claim precise, we compute the layer-visible information-theoretic (Bayes) ceiling on Has-Unknown probe accuracy under the 4-flag Unknown generator (atomic flags a_u, b_u, d_u, s_u each i.i.d. $\mathrm{Bernoulli}(0.15)$). The optimal binary predictor at each boundary marginalizes over flags not yet injected. Closed-form ceilings: Arith (k=2 visible flags) gives $(1 - 0.85^2) + 0.85^2 \cdot 0.85^2 = 0.7995$; Order and Set ($k$=3) give 0.9079; Logic (k=4) gives 1.0. Empirical probe accuracies of $80.0\%/91.1\%/90.8\%/99.7\%$ saturate the closed-form ceilings within sampling/rounding error (≤ 0.3pp) upstream and undershoot Logic by 0.3pp (consistent with MLP approximation of an exact 4-bit lookup).

The Has-Unknown probe is interpreted relative to its $\approx 52\%$ majority baseline (0.85^4, the prior probability of the No-Unknown class): 80.0% at the Arithmetic boundary is 28 percentage points above the majority baseline, indicating substantial encoding of input-level uncertainty even at the earliest stage. Order/Set truth values show moderate upstream decodability (∼58% vs. chance 33%) because the arithmetic output **c** is an input to both engines via bridge layers, so this is shared input context rather than downstream-task leakage. The logic operator is undecodable before the Logic Engine (20.2% = chance for 5 classes), confirming that the architectural information bottleneck on the operator variable is faithfully respected.

Nonlinear probe control for the verdict-side bound. The verdict-side half of the uncertainty–verdict asymmetric propagation claim (§4.3) rests on upstream final-verdict probe accuracy remaining at or below the 73.4% U-vs-non-U oracle reference. A natural skeptical reading is that the information is present in the upstream representations but inaccessible to a linear classifier, in which case a more expressive probe should reveal it. To rule this out, we train a 2-hidden-layer MLP probe with architecture $128 \to 256 \to 256 \to 3$, GELU activations, and dropout 0.1 after each hidden layer, on the same 5 trained checkpoints as Table 7. Training uses AdamW (lr $= 10^{-3}$, cosine annealing), batch size 2048, 40 epochs, with an independent data seed (999) for the 50K-sample extraction and a 70/30 train/test split. We report the *best* test accuracy across all 40 epochs for each (seed, boundary) pair; this best-epoch selection deliberately biases the comparison *in the probe's favor* (and therefore *against* the verdict-side claim). Table 8 reports the comparison against the final-verdict row of Table 7.

The nonlinear probe matches the linear probe to within 0.7pp at every upstream boundary and never crosses the 73.4% U-vs-non-U oracle reference before the Logic Engine, *even though we deliberately bias the comparison in the probe's favor* through best-test-epoch selection across 40 training epochs. At the Logic boundary both probes saturate, consistent with the final verdict being already explicitly represented at that stage. We note that this does not rule out information being present in a form decodable only by much more powerful probe families (e.g., a deep network with task-specific architecture); what it rules out is the most common alternative explanation in the probing literature—that the information is linearly inaccessible but

Table 7. Mechanistic probes at domain boundaries. Linear models decode intermediate variables from 128-dim hidden vectors. Mean ± std over 5 trained checkpoints (seeds $\{42, 123, 256, 777, 999\}$); 50K samples per checkpoint. Chance level: truth value 33%, logic op 20%. Arith result measured by R^2. Standard deviations are reported to three decimal places; entries shown as 0.000 indicate std $< 5 \times 10^{-4}$, reflecting that the corresponding probe is essentially deterministic across random seeds.

Probe	Arith	Order	Set	Logic
Arith result (R^2)	0.837 ± 0.010	0.837 ± 0.021	0.811 ± 0.013	0.156 ± 0.062
Order TV (acc)	0.583 ± 0.002	1.000 ± 0.000	0.584 ± 0.001	0.973 ± 0.015
Set TV (acc)	0.588 ± 0.003	0.585 ± 0.003	0.981 ± 0.009	0.995 ± 0.003
Final verdict (acc)	0.609 ± 0.001	0.672 ± 0.002	0.697 ± 0.002	0.999 ± 0.000
Logic op (acc)	0.199 ± 0.002	0.198 ± 0.002	0.202 ± 0.004	0.884 ± 0.041
Has Unknown (acc)	0.802 ± 0.000	0.911 ± 0.000	0.908 ± 0.000	0.997 ± 0.003

Table 8. Linear vs. nonlinear probe for the final-verdict variable at each domain boundary. The nonlinear probe is a 2-hidden-layer MLP ($128 \to 256 \to 256 \to 3$, GELU, dropout 0.1), trained for 40 epochs with best-test-epoch selection (biased in the probe's favor). Mean over 5 seeds; nonlinear-probe std < 0.003 across seeds. "Gap" is MLP minus linear (percentage points). The U-vs-non-U oracle reference is 73.4%. Upstream stages (Arith, Order, Set) remain at or below this reference under both probe families despite the best-epoch bias; the largest gap is +0.7pp at Set.

Boundary	Linear SVM	MLP (2-hidden)	Gap
Arith	0.609	0.611	+0.2pp
Order	0.672	0.676	+0.4pp
Set	0.697	0.704	+0.7pp
Logic	0.999	1.000	+0.0pp

sits just under the surface of a standard nonlinear probe.

Table 9. Layer-wise linear probing of the 8-layer Transformer baseline (Table 1), analogous to Table 2. Mean ± std over 5 seeds; 50K samples per seed. F–T distances are Euclidean; post-LN normalization may contribute to the layer-1 contraction.

Layer	**SVM Accuracy**	**F–T Distance**
input embed.	0.750 ± 0.001	0.381 ± 0.011
layer 1	0.781 ± 0.007	0.224 ± 0.025
layer 2	0.785 ± 0.011	0.342 ± 0.097
layer 3	0.859 ± 0.040	1.305 ± 0.514
layer 4	0.999 ± 0.000	8.322 ± 3.113
layer 5	1.000 ± 0.000	13.137 ± 0.479
layer 6	1.000 ± 0.000	14.470 ± 0.767
layer 7	1.000 ± 0.000	14.841 ± 0.694
layer 8	1.000 ± 0.000	16.036 ± 0.928
Layer 8 / input ratio	$\mathbf{42.1 \pm 2.5}\times$	

Layer-wise probing of the Transformer baseline. For direct comparison, we apply the same probing protocol layer-by-layer to the (larger) 8-layer Transformer baseline of Table 1, which reaches the same final correctness as THEIA on the Kleene diagnostic. The probing analysis uses the development-default BigTransformer checkpoints of Table 1 (§4.2); whether the contraction–expansion trajectory persists under the tuned configuration of Appendix H is left to future work. The trajectory (Table 9) differs qualitatively from THEIA's boundary-localized contraction–expansion pattern.

Operator-identity decomposition of the Set-boundary non-Unknown lift. The uncertainty–verdict propagation analysis of §4.3 reports that the Set-boundary verdict probe reaches 60.13% on the non-Unknown (NU) subset, +4.09pp above the mixed-operator NU majority of 56.04%. The U-vs-non-U oracle reference derived in the main text does not apply on the NU subset (it implicitly uses the Has-Unknown signal, which is constant = 0 here), so we require a separate explanation for this lift. This appendix provides the decomposition summarized in the main text.

We ran three controls, all on the same 5 trained checkpoints and the same 50K-sample pool (data seed 999) as Table 7, restricted to the NU subset (approximately 8,972 samples per seed). All probes use `LinearSVC` with `StandardScaler`, regularization C=1.0, and a 70/30 train/test split; final-verdict probes are 3-class, and the operator-identity probe is 5-class (5 logic operators).

Probe A (replicates the main-text Set-boundary NU probe) uses the 128-dim Set-boundary hidden vector as input and the 3-class verdict as target: 60.13% ± 0.33% (+4.09pp above 56.04%).

Probe B (operator-only baseline) uses the 5-dim one-hot logic operator as the sole input and the verdict as target: 48.40% ± 0.00% (deterministic given the one-hot operator). This represents what a probe could achieve if it had perfect operator knowledge but no other information.

Probe C (Set hidden + operator) uses the concatenation of Set-boundary hidden and the operator one-hot (133-dim): 61.93% ± 0.16%. The marginal gain from adding operator information on top of the Set hidden is only +1.80pp, indicating that the Set hidden already carries almost all the information relevant to Probe A.

Probe D (direct operator-identity probe) uses the Set-

boundary hidden vector as input and the 5-class logic operator as target: **20.25% ± 0.38%**, indistinguishable from 20% chance. This is the decisive test: the Set boundary encodes no operator identity.

Table 10 extends Probe D to all three upstream boundaries, showing that the operator is not decodable from Arith, Order, or Set representations.

Table 10. Operator-identity probe (Probe D) at all three upstream boundaries. Direct 5-class `LinearSVC` from boundary hidden to the logic operator label; 5 seeds, NU subset, 50K samples, 70/30 split. All three boundaries are indistinguishable from 20% chance: no upstream boundary encodes operator identity.

Boundary	Probe D acc (%)	Δ vs. chance	Verdict
Arith	19.88 ± 0.16	−0.12pp	at chance
Order	19.78 ± 0.25	−0.22pp	at chance
Set	20.25 ± 0.38	+0.25pp	at chance

Having excluded operator-identity encoding, we test whether the +4pp lift reflects per-operator verdict leakage. Stratifying the NU subset by logic operator and re-running the Set-boundary verdict probe within each operator group yields per-operator accuracies that sit *below* each operator's NU verdict majority by 6–7pp (Table 11)—the opposite direction of what leakage would produce.

Table 11. Per-operator Set-boundary verdict probe on the NU subset, compared to the per-operator majority baseline. The probe accuracy falls below the majority baseline for every operator except IFF (where both are $\approx 50\%$ by definition). There is no operator under which the probe exceeds what the NU class distribution alone would afford, ruling out per-operator verdict leakage. Values are mean ± std over 5 seeds.

Operator	Probe C acc (%)	Per-op NU majority (%)	Δ (pp)
AND	72.08 ± 0.09	79.16	−7.08
OR	72.68 ± 0.22	78.89	−6.21
IMP	71.08 ± 0.30	78.38	−7.29
IFF	51.32 ± 0.70	50.19	+1.13

The mixed-operator +4pp lift is therefore neither operator-identity encoding (Table 10) nor per-operator verdict leakage (Table 11); it is fully accounted for by val_{set}'s Kleene-correlation with the verdict conditional on the operator. val_{set} is the designated output of the Set Engine by construction; because Kleene operators such as OR have absorbing elements ($T \vee X = T$), the distribution of the verdict conditional on $(\text{val}_{\text{set}}, \text{operator})$ is non-uniform, and a mixed-operator probe can exploit this conditional structure to beat the global NU majority without encoding either the verdict or the operator directly. The three-layer persistence structure—uncertainty, designated intermediate variables, verdict—is preserved: upstream boundaries encode uncertainty and each engine's designated variable, and the final verdict appears only at the Logic Engine boundary.

F. Causal Verification via Activation Patching: Construction Details

This appendix provides the full construction protocol, baseline-correctness intermediate counts, and per-seed eligibility numbers for the activation-patching results summarized in §4.3.

OR pair construction. We construct 1,000 matched $(T \vee F,\ U \vee F)$ pairs on a converged seed-42 checkpoint, with shared a, b, arithmetic operator, set bits (chosen so $c \notin S$, fixing $\text{val}_{\text{set}} = F$), logic operator (OR), and definite flags, so that $\mathbf{c}$, both bridge outputs, and $\mathbf{v}_{\text{set}}$ are identical across sides by construction (equality verified numerically; see the AND paragraph below). The two sides differ only in the Order-Engine inputs: T-side sets $d = \max(0, c-1)$ with d_{unk}=False under REL_GTE (so $\text{val}_{\text{ord}}=T$); U-side sets d_{unk}=True (so $\text{val}_{\text{ord}}=U$). Expected Kleene outputs are $T \vee F = T$ and $U \vee F = U$. For each pair we then classify $\text{OutHead}(\text{LogicEngine}(\mathbf{v}_{\text{ord}}^{(U)}, \mathbf{v}_{\text{set}}^{(T)}, \text{OR}))$, feeding the U-side ord vector into the otherwise-T-side forward pass.

Baseline correctness and eligibility. Under data seed 12345 for pair construction, baselines are correct on all $1000/1000$ T-side pairs and on 986/1000 U-side pairs—the residual U-side error is consistent in direction with the single-checkpoint noise around the 99.85% U∨F per-rule accuracy reported in Appendix D—and we restrict the patching analysis to the 986 pairs where both baselines classify correctly. The patched prediction flips from T to U on 986/986 $= 100.0\%$ of pairs, with zero instances remaining at T and zero falling to F.

Multi-seed extension ($n=5$). Across all 5 trained checkpoints (seeds $\{42, 123, 256, 777, 999\}$), aggregate flip rate is 4898/4898 $= 100.0\%$, with no per-seed flip rate below 100%. The minimum per-seed eligible count is 960 (driven by T-side baseline correctness on the corresponding pair construction), demonstrating that causal localization at the Logic-Engine boundary is robust across seeds.

AND replication. To rule out OR-specific localization, we replicate the protocol with logic operator AND on matched $(T \wedge T,\ U \wedge T)$ pairs constructed so that $\mathbf{v}_{\text{set}}$ is identical across sides ($c \in S$, set_unk = False, fixing $\text{val}_{\text{set}}=T$). Patching $\mathbf{v}_{\text{ord}}$ from the U-side into the T-side forward pass flips the prediction from T to U on 4719/4719 $= 100.0\%$ of eligible pairs, replicated across the same 5 seeds (per-seed minimum 100.0%; minimum per-seed eligible count 920). The $\mathbf{v}_{\text{set}}$ equality check (`torch.allclose`, atol 10^{-6}) passes on all 4719/4719 AND pairs. Causal mediation by $\mathbf{v}_{\text{ord}}$ at the Logic Engine boundary therefore generalizes across both AND and OR operators on the non-absorbent configurations tested.

G. Backbone Ablation: Structured Bias Is Required for Compositional Generalization

To isolate the contribution of THEIA's four-engine modular factorization from the rest of the architecture, we conduct three controlled ablations. We replace the step computer while keeping *everything else identical*: the input encoders, the transition network, the Gumbel-softmax straight-through discretization, the three-phase training protocol of §4.5, the data generation, and the class weights. The three variants are: (1) a small flat MLP (hidden 512, 3 layers, $\sim$0.80M parameters); (2) a larger flat MLP (hidden $\sim$1024, 3 layers, $\sim$2.75M parameters, $\approx 1.8\times$ larger than THEIA's 1.51M chain step, ruling out the hypothesis that flat-MLP collapse stems from insufficient capacity relative to THEIA); and (3) an 8-layer pre-LN Transformer variant (TF8LTuned, 3,582,147 parameters; architectural identity verified via state-dict fingerprint), trained under the identical three-phase Gumbel-softmax protocol. *Capacity asymmetry favors comparators.* The chain THEIA step is 1.51M parameters, while the 2.75M flat MLP, the 2.78M ResMLP grid, and the 3.58M Transformer are all $\approx$1.8–2.4$\times$ larger. Any length-generalization advantage reported for THEIA is therefore a lower bound on the modular architecture's contribution rather than a capacity-matched benchmark result. We note that the post-LN BigTransformer architecture of Table 1 fails to converge in the chain pipeline at the pipeline's shared learning rate (10^{-3}): Phase 1 plateaus at the class-prior level ($\sim$43%) across multiple random restarts, replicated in a clean retest. A single exploratory run at the BigTransformer lineage's own development rate (5×10^{-4}) completed three-phase training (Phase 1 at 99.9%, no restarts) but degraded to 94.8% at 500 steps; because it was a single seed with concurrent unvalidated code modifications, we do not report it as a baseline. Establishing a clean post-LN chain baseline would require architecture-specific scheduling that we leave to future work; we therefore report the pre-LN TF8LTuned variant as the attention-based baseline, which converges stably under the shared protocol. The only variable across conditions is the inductive bias of the step computer.

Results. Table 12 reports per-phase training accuracy and length-generalization eval across 5 seeds $\{42, 123, 256, 777, 999\}$.

Interpretation. Four findings emerge.

(1) Capacity is not the bottleneck. The larger 2.75M flat MLP (1.8$\times$ THEIA's chain-step capacity) collapses to the same chance-level basin ($\sim$33% at 500 steps) as the smaller 0.80M variant; both match Phase 1 local accuracy to within 0.04%. The collapse is therefore caused by the flat MLP's inability to maintain local accuracy under Phase 3 Gumbel-softmax end-to-end training, not by insufficient capacity relative to THEIA.

(2) Residual connections are mean-accuracy-sensitive to depth but unreliable at the $\geq$99% threshold across the tested grid. We probe the residual-architecture design space along two axes—depth ($\in\{4, 8\}$ blocks) $\times$ expansion ratio ($\in\{2d, 4d\}$)—at a fixed parameter budget of 2.77–2.85M. Each ResMLP block is `Linear(`$d \to ed$`)` $\to$ `GELU` $\to$ `LayerNorm` $\to$ `Linear(`$ed \to d$`) + skip` with $e \in \{2, 4\}$; hidden dimension d is solved per configuration to match parameter budget (details in Table 13). Every configuration matches THEIA's Phase 1 local accuracy within 0.03pp and survives past 50 steps where flat MLPs collapse (e.g., 4b$\times$4d: 98.00% at 50 steps vs. 34.36% for the matched flat MLP). Table 13 reports per-configuration 500-step accuracy across 5 seeds $\{42, 123, 256, 777, 999\}$.

Strict 4-seed no-restart aggregate (single-try Phase 1): excluding the one auto-restart trigger (4b$\times$4d seed 999), $2/19$ (config, seed) trials reach $\geq 99\%$. *As-specified aggregate (auto-restart engaged):* $3/20$ trials reach $\geq 99\%$. Both framings support the "unreliable" finding: no single ResMLP configuration achieves a majority of seeds at $\geq 99\%$, in contrast to THEIA's 5/5. Fisher's exact test against THEIA yields $p \approx 1.1 \times 10^{-3}$ (as-specified) and $p \approx 4.9 \times 10^{-4}$ (strict).

Sub-finding: depth dominates expansion within the tested grid. Holding parameter budget fixed, doubling depth from 4 to 8 blocks raises mean 500-step accuracy by $\approx$ 11pp (4-block configs: 86.15–87.49%; 8-block configs: 97.95–97.97%), while changing expansion from $2d$ to $4d$ at fixed depth is nearly neutral (+1.34pp at depth 4, -0.02pp at depth 8). The two 8-block configurations are statistical twins at the mean-accuracy level despite an 80-unit difference in hidden dimension (d=276 vs d=198), suggesting that within this parameter budget expansion ratio is a weak lever and depth is the dominant one. Neither depth doubling nor expansion change is sufficient to cross the $\geq 99\%$ reliability threshold at 5-seed granularity.

Sub-finding: seed $\times$ architecture interaction. Seed 42 reaches $\geq$ 99% on both 8-block configurations (99.27% at 8b$\times$2d, 99.12% at 8b$\times$4d) but fails on both 4-block configurations (90.49% at 4b$\times$4d, 53.27% at 4b$\times$2d). This is not noise-level variability: the 4b$\times$2d seed-42 run reaches only 53.27%, a catastrophic failure 34pp below the next-worst 4b$\times$2d seed. The residual family has predictable seed$\times$depth structure rather than i.i.d. seed noise at the $\geq 99\%$ threshold, which the aggregate $3/20$ count captures but which any single-configuration 5-seed study would mischaracterize.

Scope of the "residual-only unreliable" claim. The probe covers a 2×2 grid of 4 configurations at a single parameter scale (2.77–2.85M). We did not sweep deeper architectures (≥ 12 blocks), wider expansions ($\geq 8d$), pre- vs. post-LN placement, or DenseNet-style multi-scale residual variants.

Table 12. Backbone ablation: five step-computer architectures under the identical three-phase Gumbel-softmax protocol. Both flat-MLP variants match THEIA on local accuracy (Phase 1) but collapse under end-to-end discretized training (Phase 3). The ResMLP column reports the 4-block $\times$ $4d$-expansion baseline of the 2×2 depth$\times$expansion grid; the complete 4-configuration grid is in Table 13. The pre-LN TF8LTuned Transformer sustains length generalization. Mean $\pm$ std over 5 seeds for THEIA, MLPs, and TF8LTuned (the latter per main paper); the ResMLP $4\times4d$ entry shows the 4-seed no-restart mean (excluding seed 999 which triggered the Phase 1 auto-restart), with the 5-seed as-specified alternative ($86.15\% \pm 9.17\%$, $1/5$ reach $\geq 99\%$) in Table 13. The TF8LTuned column reports the 8-layer pre-LN variant (3,582,147 parameters); the post-LN BigTransformer of Table 1 fails to converge in the chain pipeline at the shared learning rate 10^{-3} (see introductory paragraph above).

	THEIA (1.51M)	MLP (0.80M)	MLP (2.75M)	ResMLP (2.85M)	TF8LTuned (3.58M)
Phase 1	99.94%	99.91%	99.90%	99.91%	99.92%
Phase 2	100%	100%	100%	100%	100%
5-step	99.97	80.44	76.88	99.76	99.99
10-step	99.97	66.92	61.85	99.58	99.98
50-step	99.97	35.13	34.36	98.00	99.94
100-step	99.97	33.32	33.54	96.05	99.86
500-step	**99.96 $\pm$ 0.04**	33.58	33.35	82.83 ± 6.22	**99.24 $\pm$ 0.34**

Table 13. ResMLP 2×2 depth$\times$expansion grid at 500 steps. All four configurations target the same parameter budget (2.77–2.85M); hidden dimension d solved per configuration. Seeds $\{42, 123, 256, 777, 999\}$ identical to THEIA and the 8L Transformer baseline. The "$\geq$99%" column reports the count of seeds reaching the reliability threshold (as-specified, including auto-restart where triggered).

Config	Params	500-step (mean $\pm$ std)	Range	$\geq$99%
4b $\times$ 4d (d=280)	2,848,547	$86.15 \pm 9.17^{\dagger}$	75.27–99.41	1/5†
4b $\times$ 2d (d=383)	2,776,047	87.49 ± 19.16	53.27–97.71	0/5
8b $\times$ 2d (d=276)	2,774,659	97.97 ± 1.43	95.55–99.27	1/5
8b $\times$ 4d (d=198)	2,780,707	97.95 ± 2.03	94.34–99.12	1/5
Aggregate (as-specified): $3/20$ *trials* $\geq 99\%$				
Aggregate (strict, excluding 4b$\times$4d seed 999 auto-restart): $2/19$				

†4b$\times$4d seed 999 triggered the auto-restart specified in §4.5 ($\sim$2$\times$ Phase 1 compute) and reached 99.41%; the strict 4-seed aggregate is 82.83 ± 6.22, $0/4 \geq 99\%$.

The claim "no tested ResMLP configuration achieves a majority of seeds at $\geq 99\%$" is scoped to this grid. A positive result at ≥ 12 blocks or with pre-LN placement would refine but not refute the claim, since the observed depth effect (+11pp at $4 \rightarrow 8$ blocks) extrapolated naively would still leave 12-block configurations below the $\geq 99\%$ reliability threshold without a qualitative regime change. Complete grid coverage is deferred to the archival version of this work.

(3) Structured architectures (modular or attention-based) are reliable. Both THEIA ($99.96\% \pm 0.04\%$, 5/5 seeds in the 99.90–99.99% range) and the pre-LN TF8LTuned Transformer ($99.24\% \pm 0.34\%$, 3/5 seeds $\geq$ 99%) sustain length generalization at 500 steps, albeit at different reliability levels. The cross-seed reliability difference is sharp: THEIA's std (0.04%) is roughly 9$\times$ tighter than the Transformer's (0.34%), and across the entire ResMLP grid THEIA is $\approx 36\times$ tighter than the best ResMLP configuration (8b$\times$2d, σ=1.43%) and $\approx$ 479$\times$ tighter than the worst (4b$\times$2d, σ=19.16%), indicating that domain-separated structure contributes not only to mean accuracy but to training-stochasticity robustness in a way no tested residual configuration matches.

(4) THEIA retains a quantitative edge over the Transformer. THEIA's 500-step accuracy (99.96%) exceeds the TF8LTuned Transformer's (99.24%) by 0.72pp, and all 5 THEIA seeds lie in the 99.90–99.99% range vs. 5 of 5 Transformer seeds below 99.9% (TF8LTuned 500-step range: 98.85–99.66%). This complements the mechanistic interpretability findings of §4.3: the modular structure provides both inspectability and a tighter compositional-generalization margin under discretized composition.

Summary. The ablation reveals a reliability spectrum rather than a binary: flat MLPs collapse ($\sim$33%); ResMLP variants across a 2×2 depth$\times$expansion grid (Table 13) partially recover with a strong depth effect (4-block mean 86–87% vs 8-block mean $\sim$98%) but only $3/20$ (config, seed) trials reach $\geq$ 99% ($2/19$ strict); both modular (THEIA, $99.96\% \pm 0.04\%$, 5/5 seeds in the 99.90–99.99% range) and attention-based (TF8LTuned, $99.24\% \pm 0.34\%$, 3/5 $\geq$99%) structures sustain reliable length generalization. THEIA's cross-seed std is roughly 9$\times$ tighter than the Transformer's and 36$\times$–479$\times$ tighter than every ResMLP configuration tested, establishing that domain-separated structure contributes cross-seed reliability above and beyond mean accu-

racy across the probed residual grid.

H. Tuned Transformer Baseline

Correction (baseline optimizer configuration). The workshop-reviewed version of this paper described the §4.2 comparison as optimizer-matched (AdamW lr=10^{-3}, cosine, batch 4096 for both architectures). A pre-release code audit, reconstructing configurations from the original run artifacts (saved per-run command lines and launcher metadata), found this description incorrect: the 8L Transformer's Kleene-task runs used the BigTransformer lineage's development configuration—AdamW (lr 5×10^{-4}, β=$(0.9, 0.999)$, weight decay 0.01), cosine annealing ($T_{\max}$=150, $\eta_{\min}$=10^{-5}), no warmup, batch 2048, gradient clip 1.0, FP16—fixed since the lineage's first script and never overridden at launch, while THEIA's runs used its own development default (lr 10^{-3}, batch 4096, $T_{\max}$=200, otherwise identical). Every reported number is unchanged; what changes is the label: the §4.2 comparison is a development-defaults comparison, not an optimizer-controlled one, and the controlled comparison is the recipe-applied-to-both setting reported below. The error traces to an orchestration script whose documentation misdescribed the configuration of the training script it invoked. We note the direction of the discrepancy: the Transformer ran at a *lower* learning rate and smaller batch than THEIA—the direction standard practice prefers for Transformers—so the development-defaults gap was not produced by forcing the Transformer onto THEIA's recipe. A residual concern is that the development configuration is still suboptimal for Transformers (no warmup, β_2=0.999, peak lr above the tuned 10^{-4}); to bound this, we rerun seed 42 of the baseline with a Transformer-standard recipe. This appendix reports the full result.

Protocol. We use the identical architecture (BigTransformer, 3,641,859 parameters), data generation (2M samples, $P_{\text{unk}} = 0.15$, 80/20 train/test split), class weights ($w_F = 1.0$, $w_T = 1.0$, $w_U = 2.0$), gradient clipping (1.0), random seed (42), CUDA determinism (`cudnn.deterministic=True`, `cudnn.benchmark=False`), and Kleene diagnostic harness as the development-default run of §4.2; the only difference is the optimizer schedule, reported as a single bundle:

- Peak learning rate: 10^{-4} (vs. 5×10^{-4} development-default)
- AdamW β: $(0.9, 0.98)$ (vs. $(0.9, 0.999)$ development-default)
- Schedule: linear warmup over 5 epochs, then cosine decay
- Weight decay: 0.01
- Batch size: 2048 (unchanged from the development-default runs)

This bundle is treated as a single "Transformer-standard" variable; we do not claim isolation of any individual hyperparameter.

Results. The tuned baseline runs for 42.4 wall-clock minutes under the 150-epoch cap. Final validation per-class accuracies are 99.79% (False), 99.90% (True), 99.99% (Unknown); the final Kleene diagnostic passes 12/12 with worst rule $U \vee T$ at 99.53%. Table 14 reports the two convergence milestones against THEIA references.

At epoch 60 all 12 Kleene rules first cross the $> 99\%$ threshold, with per-rule accuracies: $F \wedge U$ 99.80, $T \wedge U$ 100.00, $U \wedge F$ **99.06** (worst), $U \wedge T$ 99.82, $T \vee U$ 100.00, $F \vee U$ 99.87, $U \vee T$ 99.33, $U \vee F$ 99.78, $F \to U$ 99.81, $T \to U$ 100.00, $T \leftrightarrow U$ 100.00, $F \leftrightarrow U$ 100.00. At final evaluation the worst rule is $U \vee T$ at 99.53%, still well above the 99% pass threshold; all 12 rules pass.

Multi-seed extension ($n=3$). We extend the tuned protocol to seeds 123 and 256 using the same architecture (BigTransformer) and the same optimizer schedule as seed 42. Both reach ≥99.93% / ≥99.90% overall accuracy and pass 12/12 Kleene rules. Seed 123 reaches overall ≥99.9% at epoch 73 (34.2 min); seed 256 at epoch 98 (43.3 min). Across the three seeds, overall-99.9% convergence time is 39.7 ± 5.0 minutes, with 12/12 Kleene coverage reached in 28.8 ± 1.7 minutes. Notably, seed 123—the only seed that fails under the development-default configuration—converges readily under tuning, confirming that its failure reflects optimizer sensitivity rather than an architectural limitation.

Transformer-recipe-applied-to-both control. The main-text wall-clock comparison (§4.2) uses THEIA at its development default (AdamW lr= 10^{-3}) against the Transformer at its development default and, separately, at the Transformer-standard tuned recipe. As a partial control we apply the same Transformer-standard recipe to THEIA (peak lr $= 10^{-4}$, AdamW $\beta = (0.9, 0.98)$, linear warmup 5 epochs then cosine, weight decay 0.01, batch size 2048, gradient clipping 1.0), obtaining 12/12 Kleene coverage across 5 seeds $\{42, 123, 256, 777, 999\}$ in 9.88 ± 1.42 minutes per seed (per-seed range $[7.71, 11.40]$ min). The Transformer under the same recipe (n=5, same seeds) converges at 48.68 ± 2.98 minutes, yielding a wall-clock ratio of $4.93\times$ (Welch two-sample unpaired statistic, 95% non-parametric bootstrap CI $[4.40, 5.66]$ from 1000 resamples at RNG seed 42; paired per-seed auxiliary: $4.99\times$, CI $[4.55, 5.49]$; per-seed pairwise ratios $\{4.68, 5.12, 4.87, 4.32, 5.97\}$). This positions between the development-defaults $6.5\times$ and the Transformer-only-tuned $3.6\times$. *Asymmetry caveat and denominator note.* The recipe slows THEIA from 7.93 to

Table 14. Tuned 8L Transformer baseline, seed 42: convergence-time milestones vs. THEIA references. Kleene 12/12 is the primary comparison point (matches §4.2); the overall-99.9% row exhibits an opposite ordering discussed in the interpretation below. Seeds 123 and 256 are reported separately in the text.

Milestone	Tuned TF8L	THEIA ref	Ratio
Kleene 12/12	**28.9 min** @ ep 60	7.93 min	$\sim\mathbf{3.6}\times$
Overall $\geq 99.9\%$	41.7 min @ ep 84	5.7 min	$\sim 7.3\times$

9.88 min ($\sim 25\%$ degradation), placing THEIA at a sub-optimal configuration while the Transformer runs near its own optimum; the $3.6\times$ and $4.93\times$ ratios share the same Transformer runtime but differ in the THEIA baseline (development-default vs recipe-applied), so the $4.93\times$ 95% CI $[4.40, 5.66]$ does not overlap $3.6\times$ by denominator shift rather than stochastic disagreement. The $4.93\times$ ratio is therefore read as: *even when THEIA is forced onto the Transformer's preferred recipe, the Transformer's own preferred recipe does not close the wall-clock gap*. A genuinely symmetric comparison (each architecture at its own optimum) remains future work.

Full 39-rule standard K3 coverage ($n=3$). We extend the post-hoc diagnostic from the 12 targeted Unknown-involving rules (Table 5) to the complete 39-rule Kleene K3 truth table (Table 6: 36 binary rules spanning $\{\wedge, \vee, \rightarrow, \leftrightarrow\} \times \{F, T, U\}^2$, plus 3 unary NOT rules; 10,000 samples per rule per seed). Standard strong Kleene ground truth (IMP with $U \rightarrow T = T$, $U \rightarrow F = U$) is imported directly from the Transformer training script to rule out any diagnostic–training semantics mismatch. All three tuned BigTransformer checkpoints pass 39/39 at $> 99\%$ on every seed (117/117 rule–seed combinations; grand mean 99.89%; worst single combination $F \leftrightarrow T$ at 99.48% on seed 42). Per-seed worst rules are $F \leftrightarrow T$ (seed 42, 99.48%) and $T \rightarrow F$ (seeds 123 and 256, both at 99.50%); all three are non-Unknown-involving and therefore not in the 12 targeted Unknown rules of Table 5. The 3 NOT rules are uniformly strong ($\geq 99.87\%$). This diagnostic shares ground truth, coverage, and pass threshold with Table 6 (THEIA, $n=5$), providing a matched full-coverage comparison across architectures: both THEIA and the tuned BigTransformer learn complete standard Kleene K3 at $> 99\%$ per-rule accuracy on every rule and every seed tested.

Interpretation and the overall-vs-Kleene discrepancy. The tuned baseline (seed 42) reaches 12/12 Kleene coverage *before* it reaches overall 99.9% validation accuracy (epoch 60 vs. epoch 84, 28.9 min vs. 41.7 min). This is the opposite ordering from the development-default Transformer, which reaches overall 99.9% first and requires additional training to stabilize all 12 Kleene rules. The most plausible explanation is that the low-lr tuned recipe stabilizes the rare Kleene edge cases earlier (because smaller update steps are less likely to oscillate on high-variance low-frequency configurations), while the final $\sim 0.1\%$ of overall accuracy—dominated by bulk distribution fitting—takes proportionally longer. A practical consequence: *the overall-99.9% milestone is a misleading comparison point for tuned Transformer baselines*, because the tuned recipe trades bulk-accuracy optimization speed for Kleene-rule stability. The Kleene-aware milestone of the main result (§4.2) is the meaningful operating point for this task, and the $6.5\times \rightarrow \sim 3.6\times$ narrowing under tuning is the meaningful quantitative update.

Caveats. (1) The 5 tuned-package hyperparameters (peak lr, β, warmup schedule, weight decay, batch size) are changed jointly; we do not isolate which component is most responsible for the narrowing. (2) Exhaustive hyperparameter search (different warmup lengths, alternative schedules, different peak lr) was not performed; a more aggressive tuned configuration could in principle narrow the gap further. (3) The layer-wise probing analysis in §4.3 and Table 9 uses the development-default Transformer checkpoints; whether the contraction–expansion trajectory persists under the tuned configuration reported here is left to future work. Despite these caveats, the qualitative conclusion is robust: Transformer-specific tuning narrows but does not eliminate the convergence-time gap, and the modular architecture retains a meaningful wall-clock advantage under both comparison regimes.

I. Reproducibility

All experiments use PyTorch with CUDA on a single NVIDIA RTX 5080. Random seeds for multi-seed experiments: $\{42, 123, 256, 777, 999\}$; Transformer baseline extended to 8 seeds with $\{31415, 27182, 14142\}$; tuned protocol extended to 3 seeds $\{42, 123, 256\}$. Training hyperparameters: THEIA, AdamW lr 10^{-3}, weight decay 0.01, cosine annealing, batch 4096, FP16; Transformer Kleene baseline, AdamW lr 5×10^{-4}, weight decay 0.01, cosine ($T_{\max}$=150), batch 2048 (Correction in App. H). Four-domain training: 2M samples, batch size 4096, 80–120 epochs depending on early-stopping criterion ($\sim$8 minutes per seed on average). Multi-hop training: 1M samples, batch size 2048, $\sim$50 epochs ($\sim$15 minutes). Sequential chain training: three-phase pipeline, 2M single-step + 500K chain samples, $\sim$16 minutes per seed; Phase 3 Gumbel-softmax temperature schedule $\tau = \max(0.1,\ 0.5 - 0.01\,\text{ep})$, decay-

ing linearly from 0.5 at epoch 1 to the floor 0.1 at epoch 40 and clamped thereafter (identical across all backbones of App. G). The Kleene diagnostic protocol used in Table 5 follows the construction described in Appendix C. The tuned Transformer follow-up of Appendix H uses the same RTX 5080 and the same data generation; only the optimizer schedule differs. Code, data generation scripts, the doubly-fixed diagnostic harness, and the complete architecture definitions of Appendix B are released under the MIT license at `https://github.com/xxx12e/theia-k3`.

References

[1] L. de Moura and N. Bjørner. Z3: An efficient SMT solver. In *TACAS*, 2008.

[2] R. Manhaeve, S. Dumančić, A. Kimmig, T. Demeester, and L. De Raedt. Neural probabilistic logic programming in DeepProbLog. *Artificial Intelligence*, 298:103504, 2021.

[3] Z. Yang, A. Ishay, and J. Lee. NeurASP: Embracing neural networks into answer set programming. In *IJCAI*, 2020.

[4] Z. Li, J. Huang, and M. Naik. Scallop: A language for neurosymbolic programming. In *PLDI*, 2023.

[5] S. C. Kleene. *Introduction to Metamathematics*. North-Holland, 1952.

[6] P. Veličković, A. Puigdomènech Badia, D. Budden, R. Pascanu, A. Banino, M. Dashevskiy, R. Hadsell, and C. Blundell. The CLRS algorithmic reasoning benchmark. In *ICML*, 2022.

[7] G. Rodionov and L. Prokhorenkova. Discrete neural algorithmic reasoning. In *ICML*, 2025.

[8] K. Xu, J. Li, M. Zhang, S. S. Du, K.-i. Kawarabayashi, and S. Jegelka. What can neural networks reason about? In *ICLR*, 2020.

[9] A. Loukas. What graph neural networks cannot learn: depth vs width. In *ICLR*, 2020.

[10] M. Fitting. A Kripke–Kleene semantics for logic programs. *J. Logic Programming*, 2(4), 1985.

[11] Q. Li, Z. Han, and X.-M. Wu. Deeper insights into Graph Convolutional Networks for semi-supervised learning. In *AAAI*, 2018.

[12] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun. Measuring and relieving the over-smoothing problem for Graph Neural Networks from the topological view. In *AAAI*, 2020.

[13] A. Trask, F. Hill, S. Reed, J. Rae, C. Dyer, and P. Blunsom. Neural arithmetic logic units. In *NeurIPS*, 2018.

[14] G. Alain and Y. Bengio. Understanding intermediate layers using linear classifier probes. In *ICLR Workshop*, 2017.

[15] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[16] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Neural module networks. In *CVPR*, 2016.

[17] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017.

[18] S. C. Chan, L.-S. Hsu, S. Brody, and H.-H. Teh. Neural three-valued-logic networks. In *International Joint Conference on Neural Networks (IJCNN)*, 1989.

[19] L.-S. Hsu, H.-H. Teh, S.-C. Chan, and K.-F. Loe. Multi-valued neural logic networks. In *Proc. 20th International Symposium on Multiple-Valued Logic (ISMVL)*, pages 426–432, 1990.

[20] G. Wang and H. Shi. TMLNN: Triple-valued or multiple-valued logic neural network. *IEEE Transactions on Neural Networks*, 9(6):1099–1117, 1998.

[21] R. Riegel, A. Gray, F. Luus, N. Khan, N. Makondo, I. Y. Akhalwaya, H. Qian, R. Fagin, F. Barahona, U. Sharma, S. Ikbal, H. Karanam, S. Neelam, A. Likhyani, and S. Srivastava. Logical Neural Networks. *arXiv:2006.13155*, 2020.

[22] G. Marra, S. Dumančić, R. Manhaeve, and L. De Raedt. From statistical relational to neurosymbolic artificial intelligence: A survey. *Artificial Intelligence*, 328:104062, 2024.

[23] L. Serafini and A. S. d'Avila Garcez. Logic Tensor Networks: Deep learning and logical reasoning from data and knowledge. In *Proc. NeSy Workshop*, 2016.

[24] S. Badreddine, A. S. d'Avila Garcez, L. Serafini, and M. Spranger. Logic Tensor Networks. *Artificial Intelligence*, 303:103649, 2022.

[25] J. Xu, Z. Zhang, T. Friedman, Y. Liang, and G. Van den Broeck. A semantic loss function for deep learning with symbolic knowledge. In *ICML*, 2018.

[26] H. Dong, J. Mao, T. Lin, C. Wang, L. Li, and D. Zhou. Neural Logic Machines. In *ICLR*, 2019.

[27] F. Petersen, C. Borgelt, H. Kuehne, and O. Deussen. Deep Differentiable Logic Gate Networks. In *NeurIPS*, 2022. arXiv:2210.08277.

[28] F. Petersen, H. Kuehne, C. Borgelt, J. Welzel, and S. Ermon. Convolutional Differentiable Logic Gate Networks. In *NeurIPS*, 2024. arXiv:2411.04732.

[29] S. S. Damera, R. Matheu, A. G. Puranic, and J. S. Baras. Polynomial Surrogate Training for Differentiable Ternary Logic Gate Networks. arXiv:2603.00302, 2026.

[30] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with Gumbel-softmax. In *ICLR*, 2017.

[31] C. J. Maddison, A. Mnih, and Y. W. Teh. The Concrete distribution: A continuous relaxation of discrete random variables. In *ICLR*, 2017.

[32] Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv:1308.3432*, 2013.

[33] S. Mittal, Y. Bengio, and G. Lajoie. Is a modular architecture enough? In *NeurIPS*, 2022.

[34] R. Csordás, S. van Steenkiste, and J. Schmidhuber. Are neural nets modular? Inspecting functional modularity through differentiable weight masks. In *ICLR*, 2021.

[35] B. M. Lake and M. Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *ICML*, 2018.

[36] D. Hupkes, V. Dankers, M. Mul, and E. Bruni. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795, 2020.