BEYOND LABELS: PAC LEARNING WITH INTERMEDIATE SUPERVISION

Anonymous authors
Paper under double-blind review

ABSTRACT

The traditional learning paradigm relies on input-label pairs but as concept classes become increasingly complex, learning becomes statistically and computationally expensive. Modern machine learning bypasses this problem using intermediate supervision like chain-of-thought and distillation to speed up learning. Although widely explored in practice, its theoretical understanding remains limited.

We study the PAC learnability of deep models composed of intermediate gates with partial information about their outputs, revealed only during training. Our flexible concept class captures a wide range of architectures, including deep ReLU networks and transformers and uses n gates in L layers to map a $d-$dimensional feature to a label. While learning is computationally intractable even for one-hidden-layer networks, we show that mild intermediate supervision makes deep models efficiently learnable.

We study different degrees of intermediate supervision ranging from exact intermediate traces to partial and noisy information. We provide tight characterizations of the computational complexity; the exact case is fully polynomial in d, n, L, while the partial and noisy cases are polynomial for constant L which we show that is unavoidable by proving a corresponding cryptographic hardness result. Finally, experiments demonstrate the benefits of intermediate supervision in cases where classical label-only approaches fail.

1 INTRODUCTION

Modern learning systems are increasingly trained with supervision that goes beyond input–output pairs. Chain-of-thought (CoT) supervision (Nye et al., 2021; Wei et al., 2022) exposes intermediate reasoning steps, knowledge distillation (Hinton et al., 2015) provides teacher-generated soft targets, and intermediate-feature supervision (Romero et al., 2015; Heo et al., 2019) reveals internal representations. Empirically, such intermediate supervision can substantially aid learning, yet its effect on the computational complexity of learning remains poorly understood. We study this question for deep neural models, whose computation naturally produces a sequence of intermediate activation and routing decisions before the final prediction.

Reasoning traces describe intermediate steps in an answer, but not every task comes with such a decomposition (Phan et al., 2023; Giannoulis et al., 2025). Neural networks, however, compute internal states and routing decisions for every input. Those computations can be recorded at different levels of detail: a training example might reveal the activation values of all hidden ReLU units, which ReLU units are active, or the attention weights or which token hardmax attention selects at each position. This suggests a natural form of intermediate supervision: reveal selected parts of the target model's internal computation during training, while requiring the learner to predict from the input alone at test time.

We formalize this idea in the probably approximately correct (PAC) framework (Valiant, 2026; Blumer et al., 1989) through a teacher–student setting in which the teacher is an unknown network that computes the target function and reveals selected intermediate computations during training, while the student must predict from the input alone. This setting matters most for the cases where the main obstacle to learning is computational rather than statistical, such as deep neural networks and transformers (Vaswani et al., 2017). Polynomially many examples already suffice, in principle, to pin down a bounded-size ReLU network or certain transformer models (Bartlett et al., 2019; Yang

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

et al., 2026). But knowing that the data determine a good predictor is not the same as finding it: even for shallow ReLU networks with noiseless labels, learning from input–output pairs can be computationally hard (Daniely & Vardi, 2021; Chen et al., 2022). The information is there, but hard to extract. Seeing the teacher's intermediate computations might be the key. This motivates our central question:

Does access to intermediate computations change the computational complexity of learning?

Our answer is yes, but the effect depends on what intermediate information is revealed. Complete numerical traces can make learning polynomial in dimension, width, and depth, whereas partial traces such as activation signs or attention routes may still incur exponential dependence on depth and can remain computationally hard in general. We characterize these regimes, develop learning guarantees for exact and inexact traces, and identify conditions under which partial supervision becomes tractable.

Our contributions.

- **Complete numerical traces.** Revealed hidden states reduce learning to local linear problems (Theorem 1).
- **Partial traces.** Local learners for signs, gate values, or attention routes compose into an end-to-end learner, even with unknown gate order (Theorem 2 and Appendix E).
- **Noisy and inexact traces.** A stability theorem controls error propagation (Theorem 3).
- **Computational limits and a tractable regime.** Complete discrete traces can remain hard (Theorem 4), whereas path margins and readout norm bounds give an efficient regime (Theorem 5).
- **Experiments.** On parity, intermediate supervision enables successful learning where the labels-only gradient baseline fails (Section 5).

Related work. Intermediate supervision builds on learning from hints and learning using privileged information, where the learner receives training information unavailable at test time (Abu-Mostafa, 1990; Vapnik & Vashist, 2009). In deep learning, related ideas include auxiliary losses, concept bottlenecks, and distillation from teacher outputs, hidden representations, or activation boundaries (Lee et al., 2015; Koh et al., 2020; Hinton et al., 2015; Romero et al., 2015; Heo et al., 2019). Recent theory on CoT supervision studies when observing autoregressive reasoning traces improves learnability (Altabaa et al., 2025; Joshi et al., 2025; Hanneke et al., 2026). In contrast, we reveal variables from the target model's *internal* computation, such as ReLU gates and attention routes, rather than externally generated reasoning steps. Appendix A discusses related work in more detail.

2 Setup: Gated Linear Programs and Intermediate Supervision

We study models whose hidden states are formed by *gated linear transformations*. Let the input be $h^{(0)} = x \in \mathbb{R}^d$, and let $h^{(\ell)} \in \mathbb{R}^{d_\ell}$ denote the hidden state at layer ℓ. At layer ℓ, let k_ℓ denote the number of gated branches (e.g., neurons in a ReLU layer). The model computes

$$h^{(\ell)} = \sum_{j=1}^{k_\ell} s_j^{(\ell)} A_{\ell,j} h^{(\ell-1)}, \qquad f^\star(x) = c^\top h^{(L)}. \tag{1}$$

Here $A_{\ell,j} \in \mathbb{R}^{d_\ell \times d_{\ell-1}}$, with $d_0 = d$, is a fixed but unknown linear map, $c \in \mathbb{R}^{d_L}$ is an unknown readout vector, and $s_j^{(\ell)}$ is a scalar gate that may depend on the preceding computation. Thus $A_{\ell,j} h^{(\ell-1)}$ is the linear content of branch j, while $s_j^{(\ell)}$ determines whether, or how strongly, that branch contributes. We call a model of the form Equation 1 a *gated linear program*.

For example, a standard ReLU layer $h_j^{(\ell)} = \sigma\left(w_{\ell,j}^\top h^{(\ell-1)}\right)$, where $\sigma(t) = \max\{0, t\}$ can be written in the form of Equation 1 as

$$h^{(\ell)} = \sum_j \underbrace{\mathbb{1}\left[w_{\ell,j}^\top h^{(\ell-1)} > 0\right]}_{s_j^{(\ell)}} \underbrace{e_j w_{\ell,j}^\top}_{A_{\ell,j}} h^{(\ell-1)}.$$

Thus, the gate $s_j^{(\ell)}$ records whether neuron j is active, while $A_{\ell,j}h^{(\ell-1)}$ places its linear preactivation in the jth coordinate of the hidden state.

Beyond ReLU networks, the framework also includes:

- **Gated linear units and mixtures of linear experts.** Continuous gates, computed with their own weights, scale the linear content of each branch.
- **Residual attention.** Attention weights gate linear value transformations, with the residual connection represented by an always-on branch.

Once the gates are known, every hidden state and the final output are linear in features constructed from the input and gates alone; the unknown branch maps $A_{\ell,j}$ enter only through fixed coefficients.

Proposition 1 (Path linearization)**.** *Set $\Phi^{(0)} = x \in \mathbb{R}^d$ and, recursively, $\Phi^{(\ell)} = s^{(\ell)} \otimes \Phi^{(\ell-1)} \in \mathbb{R}^{D_\ell}$, where $u \otimes v = (u_1 v, \dots, u_k v)$ denotes the Kronecker product and $D_\ell = d\prod_{r=1}^{\ell} k_r$. Then, for every ℓ, there exists a fixed matrix $B_\ell \in \mathbb{R}^{d_\ell \times D_\ell}$ such that $h^{(\ell)} = B_\ell \Phi^{(\ell)}$. Consequently, there exists a fixed vector $\beta \in \mathbb{R}^{D_L}$ such that $f^\star(x) = \langle \beta, \Phi^{(L)} \rangle$.*

Full proof can be found in Appendix C.1. The path representation is purely algebraic and does not depend on what supervision is available to the learner.

Equation 1 specifies the teacher's computation. We next describe what parts of this computation are revealed to the learner during training.

Supervision regimes. A trace records the intermediate information revealed by the teacher while evaluating $f^\star$ on a training input. We describe a trace by three independent attributes.

1. *Coverage.* A complete numerical trace contains the hidden state at every layer. A partial trace keeps only a function of that computation, such as an activation sign, a gate value, or an attention route.
2. *Target type.* A discrete record gives a local classification problem, while a real-valued record gives a local regression problem.
3. *Observation quality.* The recorded target may be clean, corrupted, or rounded.

Section 3.1 treats complete numerical traces; Sections 3.2–3.5 exactly recoverable partial traces; Section 3.6 corrupted or rounded ones; and Section 4 asks when discrete traces suffice.

Conventions. Training examples are i.i.d. from an arbitrary input distribution $\mathcal{D}$, and labels are clean unless stated otherwise. We write $\|Z\|_{L_2} = (\mathbb{E}_{X\sim\mathcal{D}}\|Z(X)\|_2^2)^{1/2}$. A *proper* learner returns a model in the target class, while an *improper* learner may use another representation. We assume exact arithmetic, including for inverse links; running times for linear programs on rational data include the encoding length.

3 What the Teacher Reveals

3.1 Complete traces

If the teacher reveals $s^{(\ell)}$ and $h^{(\ell)}$, the covariate $s^{(\ell)} \otimes h^{(\ell-1)}$ and target $h^{(\ell)}$ are observed. Equation 1 thus gives local linear problems without the expanded path representation of Proposition 1.

Theorem 1 (Complete traces for gated linear programs)**.** *Consider a gated linear program of the form in Equation 1. Suppose that, for each training input, the teacher reveals $s^{(1)}, h^{(1)}, \dots, s^{(L)}, h^{(L)}$ and the final label. Assume that, for every layer ℓ, the gate rule $s^{(\ell)} = g_\ell(h^{(\ell-1)})$ is PAC learnable under disagreement loss, with training and prediction costs uniformly polynomial in the parameters listed at the end of the theorem.*

Then, for any $\varepsilon, \delta \in (0,1)$, there exists an improper learner that uses only the input at test time and, with probability at least $1-\delta$, satisfies

$$\Pr_x[\widehat{f}(x) \neq f^\star(x)] \leq \varepsilon.$$

Its sample complexity and running time are polynomial in L, $\max_\ell d_\ell$, $\max_\ell k_\ell d_{\ell-1}$, $1/\varepsilon$, and $\log(1/\delta)$.

Full proof is in Appendix B.1. As a concrete instance, consider ReLU networks: here $s_j^{(\ell)} = \mathbb{1}[h_j^{(\ell)} > 0]$, so activation values determine the gates. Fitting each neuron as a numerical ReLU predictor on the revealed preceding state, followed by the linear readout, gives a *proper* learner.

Corollary 1 (ReLU networks with complete activation values)**.** *A depth-L, width-at-most-n ReLU network with a linear readout is distribution-free PAC learnable from exact hidden activation values and labels. For every $\varepsilon, \delta \in (0,1)$, there is a proper learner that uses only the input at test time and, with probability at least $1-\delta$, satisfies*

$$\Pr_x[\widehat{f}(x) \neq f^\star(x)] \leq \varepsilon.$$

Its sample complexity and running time are polynomial in d, n, L, $1/\varepsilon$, and $\log(1/\delta)$.

Full proof can be found in Appendix B.2. Complete traces therefore reduce learning to a sequence of local prediction problems in the true hidden states. We now turn to partial traces, where those states are unavailable and must instead be reconstructed from the revealed intermediate information.

3.2 Partial traces: two composition mechanisms

When the numerical state is not revealed, the learner must reconstruct a usable covariate from the input and earlier partial records. Two different guarantees can support this recursion.

A gate s_q is *recovered exactly* when its learned predictor $\widehat{g}_q$, given the input and the true earlier gates $s_{<q}$, is wrong with probability at most a small tolerance α: $\Pr_{x\sim\mathcal{D}}\big[\widehat{g}_q(x, s_{<q}(x)) \neq s_q(x)\big] \leq \alpha$, where, for a vector gate, an error means failing to recover the complete vector. When gates are predicted sequentially, every later gate receives the correct covariates as long as all earlier predictions are correct. Thus, the reconstructed computation agrees with the true one until the first gate error, and the end-to-end error is controlled by a union bound over all local failures with the readout error.

For approximate recovery, working layerwise with the gate vector $s^{(\ell)}$ evaluated on the reconstructed features, the local guarantee is $\|\widehat{s}^{(\ell)} - s^{(\ell)}\|_{L_2} \leq K_\ell \|\widehat{\Phi}^{(\ell-1)} - \Phi^{(\ell-1)}\|_{L_2} + \eta$, where K_ℓ controls how much incoming feature error the gate amplifies and η is the new error it introduces. Now small errors may occur on every input and propagate through later layers, so composition requires controlling their amplification. Theorems 2 and 3 formalize these two mechanisms.

The choice between the two mechanisms is determined by the local guarantee, not by whether the revealed target is discrete or real-valued. Realizable halfspaces, invertible-link gates, numerical ReLU gates, and clean softmax rows all admit exact recovery under the assumptions used here. Approximate recovery is needed for rounded values, additive perturbations, misspecified gate classes, or any local routine that supplies only a numerical error bound.

3.3 Exact composition of partial traces

Section 3.1 reduced learning to local problems whose covariates were the revealed hidden states. A partial trace takes those covariates away: the learner sees, for instance, which neurons fired, but not the values they carried, so the next gate is no longer a halfspace in anything it observes. To keep learning locally, the learner needs a *substitute covariate*: something it can compute itself, both during training and at test time, that still carries enough information to predict the next gate.

Proposition 1 supplies a substitute covariate. Once the gates are known, every hidden state is a fixed linear function of the path feature $\Phi^{(\ell)}$, and $\Phi^{(\ell)}$ is built from the input and the gates alone. For a ReLU network, the activation sign of neuron j at layer ℓ is

$$s_j^{(\ell)} = \mathbb{1}\left[\langle w_{\ell,j}, h^{(\ell-1)}\rangle > 0\right] = \mathbb{1}\left[\langle B_{\ell-1}^\top w_{\ell,j}, \Phi^{(\ell-1)}\rangle > 0\right].$$

This is again a halfspace, but now in a feature built from x and the earlier signs, which are revealed during training and predicted by the learner at test time; the hidden state $h^{(\ell-1)}$ is never needed. Likewise, the label is linear in the final path feature, $f^\star(x) = \langle \beta, \Phi^{(L)}\rangle$.

This example contains the two ingredients we need in general: 1) each gate is learnable from features that are available *before* it is computed; and 2) the label is linear in features that are available *once the whole trace* is known. We state the result for any feature maps with these two properties.

List the revealed blocks as $s_1, \ldots, s_M$ in computation order; a block may be a scalar gate, a gate vector, or a hidden state. We call a feature map *admissible* if it can be evaluated without the teacher's parameters and uses only information available at that point in the computation. We need two kinds: for each block q, a map $U_q(x, s_{<q})$ of the input and the preceding blocks, and for the readout, a map $F(x, s_{1:M}) \in \mathbb{R}^D$ of the input and the complete trace. These maps and the learned predictors must also be defined on predicted prefixes, including erroneous ones.

Given such learners, training and prediction proceed as follows. During training, each gate predictor $\widehat{g}_q$ is fitted on covariates built from the *true* revealed prefix, and the readout $\widehat{\beta}$ is obtained by solving the linear equations $\langle \beta, F(x_i, s_{1:M}(x_i)) \rangle = f^\star(x_i)$. At test time no trace is available, so the learner rolls the computation forward using its own predictions:

$$\widehat{s}_q(x) = \widehat{g}_q\big(U_q(x, \widehat{s}_{<q}(x))\big), \quad q = 1, \ldots, M, \qquad \widehat{f}(x) = \big\langle \widehat{\beta},\, F(x, \widehat{s}_{1:M}(x)) \big\rangle.$$

The next theorem shows that if every gate is recovered exactly, the end-to-end predictor is accurate, even though at test time each gate predictor sees predicted rather than true prefixes.

Theorem 2 (Exact master theorem)**.** *Consider a gated linear program of the form in Equation 1 whose training trace reveals blocks $s_1, \ldots, s_M$ (gates or hidden states), in computation order, together with the label $f^\star(x)$. Suppose every block s_q admits an exact-recovery learner on an admissible feature map U_q, and that an admissible feature map F satisfies $f^\star(x) = \langle \beta^\star, F(x, s_{1:M}(x)) \rangle$ for all x, for some $\beta^\star \in \mathbb{R}^D$. Then, for every $\varepsilon, \delta \in (0, 1)$, there is an improper learner that uses only the input at test time and, with probability at least $1 - \delta$, satisfies*

$$\Pr_{x \sim \mathcal{D}}\big[\widehat{f}(x) \neq f^\star(x)\big] \leq \varepsilon.$$

If the feature maps are computable in polynomial time, the sample complexity and running time are polynomial in M, D, $1/\varepsilon$, and $\log(1/\delta)$, together with the costs of the local gate learners.

Proof sketch. Each gate predictor $\widehat{g}_q$ fails on its true covariate $U_q(x, s_{<q}(x))$ with probability at most α_q, and the fitted readout $\widehat{\beta}$ fails on the true feature with probability at most α_{out}. Outside the union of these events, of probability at most $\sum_q \alpha_q + \alpha_{\mathrm{out}}$, each gate receives its true covariate and is predicted correctly, so by induction the readout sees the true feature and returns $f^\star(x)$. Choosing the local accuracies so that this sum is at most ε gives the theorem. Full proof is in Appendix C.3.

3.4 Clean partial traces: classification and regression gates

We now apply Theorem 2 to concrete architectures. By the theorem, it suffices to give an exact-recovery learner for each gate on admissible features. For sign gates this is a classification problem, and for numerical gates a regression problem.

ReLU activation signs. Suppose the trace reveals whether each neuron is active. As shown in Section 3.3, the sign $s_j^{(\ell)}$, which is used to build $\Phi^{(\ell)}$, is a halfspace in $\Phi^{(\ell-1)}$, and the output $f^\star(x)$ is linear in $\Phi^{(L)}$. Since a consistent halfspace can be found by linear programming (LP), Theorem 2 applies with $U_q = \Phi^{(\ell-1)}$ and $F = \Phi^{(L)}$ and gives the following result.

Corollary 2 (ReLU networks)**.** *A depth-L, width-at-most-n, bias-free ReLU network with a linear readout is distribution-free PAC learnable from noiseless ordered sign gates and labels, with $O\big(\frac{nL}{\varepsilon}\big(dn^{L-1}\log\frac{2nL}{\varepsilon} + \log\frac{2nL}{\delta}\big)\big)$ examples. The learner runs in time polynomial in the path dimension dn^L.*

Full proof is in Appendix C.5. The sign trace thus removes the need to learn the network parameters directly, but not the path expansion: since $D_L = dn^L$, the result is polynomial only for fixed depth. This is the cost of this generic path representation. With complete activation values (Corollary 1), the local covariate is the hidden state itself and the dependence on depth is polynomial. In contrast, Theorem 4 shows that from sign traces alone, no learner is polynomial in depth in general.

Unknown gate order. The result extends to sign gates revealed under a fixed but unknown permutation, so the learner need not know which gate or layer each revealed sign belongs to. The resulting learner is polynomial in $d2^{nL}$, $1/\varepsilon$, and $\log(1/\delta)$; see Theorem 6 in Appendix E.

Clean numerical gates. The same framework applies to real-valued gates. If a gate has the form $s(U) = \psi(BU)$ with a known injective link ψ, applying ψ^{-1} to the observed value gives the linear target BU, so B can be fitted by solving a linear system and the gate is recovered exactly. This covers, for example, sigmoid-gated linear units. Softmax gates require a separate argument, given in the next section, and further examples are in Appendix C.7.

3.5 Residual attention

For attention, the generic path representation is far from the most economical choice. Consider an L-layer, single-head self-attention model on N input tokens $x_1, \dots, x_N \in \mathbb{R}^d$, with $h_t^{(0)} = x_t$. At layer ℓ, let $W_\ell \in \mathbb{R}^{d\times d}$ determine the attention scores and let $A_\ell \in \mathbb{R}^{d\times d}$ be the linear map applied to the attended values; in standard query–key–value notation, $W_\ell = W_Q^\top W_K/\sqrt{d_k}$ and $A_\ell = W_O W_V$. For tokens $t, i \in [N]$, define $b_{\ell,t,i} = (h_t^{(\ell-1)})^\top W_\ell h_i^{(\ell-1)}$. The attention weights are either $p_{\ell,t,i} = \exp(b_{\ell,t,i})/\sum_{r=1}^N \exp(b_{\ell,t,r})$ for softmax attention, or $p_{\ell,t,i} = \mathbb{1}\big[i = \arg\max_{r\in[N]} b_{\ell,t,r}\big]$ for hardmax attention, assuming a unique maximizer. Token representations are then updated by

$$h_t^{(\ell)} = h_t^{(\ell-1)} + A_\ell \sum_{i=1}^N p_{\ell,t,i} h_i^{(\ell-1)}, \tag{2}$$

and we use a linear readout from $h_N^{(L)}$. Our framework covers essentially all components of a Transformer block except layer normalization.

A compact lift. Because attention mixes token representations with the same linear map A_ℓ for every source token, each hidden state stays linear in a small feature built from the input and the attention weights. Define

$$z_t^{(0)} = x_t, \qquad z_t^{(\ell)} = \Big(z_t^{(\ell-1)}, \sum_i p_{\ell,t,i}\, z_i^{(\ell-1)}\Big) \in \mathbb{R}^{d\,2^\ell}, \tag{3}$$

which keeps the previous representation of token t together with the attention-weighted average formed at layer ℓ. Then $h_t^{(\ell)} = B_\ell z_t^{(\ell)}$ with $B_0 = I_d$ and $B_\ell = [B_{\ell-1} \;\; A_\ell B_{\ell-1}]$. The feature is admissible, and its dimension doubles with each layer but does not depend on N.

Substituting into the scores gives $b_{\ell,t,i} = (z_t^{(\ell-1)})^\top C_\ell\, z_i^{(\ell-1)}$ for a fixed unknown matrix C_ℓ (all reductions below are detailed in Appendix C.6), so each score is linear in $z_t^{(\ell-1)} \otimes z_i^{(\ell-1)}$. For hardmax attention, predicting the route of token t is then a multiclass linear classification problem. For softmax attention, a row determines its scores only up to a common shift, but the log-ratios $\log(p_{\ell,t,i}/p_{\ell,t,N}) = b_{\ell,t,i} - b_{\ell,t,N}$ are linear in C_ℓ and determine the complete row. Theorem 2 applies with $U_{(\ell,t)} = (z_t^{(\ell-1)} \otimes z_i^{(\ell-1)})_{i=1}^N$ and $F = z_N^{(L)}$.

Corollary 3 (Residual attention)**.** *Consider the L-layer residual attention model in Equation 2 with N tokens and a linear readout. From noiseless complete hardmax routes or noiseless complete softmax attention rows, together with labels, the model is distribution-free PAC learnable by an improper learner that uses only the input at test time. For hardmax attention, it suffices to use $O\big(\frac{LN}{\varepsilon}\big(d^2\, 4^{L-1}\log\frac{2NL}{\varepsilon} + \log\frac{2LN}{\delta}\big)\big)$ examples, and for softmax attention $O\big(\frac{LN}{\varepsilon}\big(d^2\, 4^{L-1} + \log\frac{2LN}{\delta}\big)\big)$ examples, plus $O\big((d\,2^L + \log(1/\delta))/\varepsilon\big)$ examples for the readout. The algorithms run in time polynomial in d, N, 2^L, $1/\varepsilon$, and $\log(1/\delta)$.*

As with ReLU signs, the dependence on depth remains exponential. For hardmax routes, Theorem 4 shows that this cannot be avoided in general. Full proof can be found in Appendix C.6.

3.6 Inexact real-valued traces

The exact results rely on the first-mistake argument, which excludes a small set of bad inputs. When traces are rounded or noisy, small errors occur on *every* input and feed into all later features, so this argument breaks down. We therefore follow the approximate mechanism of Section 3.2: we track how the error of the reconstructed features grows from layer to layer, and measure accuracy in L_2 rather than by disagreement probability.

We work with the path features of Proposition 1, and write $\widehat{\Phi}^{(\ell)}$ for the feature obtained by substituting predicted gates for true ones. Since each gate is learned from reconstructed features, we ask for a local guarantee that allows its error to depend on the error of the features it receives:

$$\|\widehat{s}^{(\ell)} - s^{(\ell)}\|_{L_2} \le K_\ell \|\widehat{\Phi}^{(\ell-1)} - \Phi^{(\ell-1)}\|_{L_2} + \eta. \tag{4}$$

Since $\Phi^{(\ell)}$ is built from $s^{(\ell)}$ and $\Phi^{(\ell-1)}$, the feature error at layer ℓ then consists of the error inherited from layer $\ell-1$, possibly amplified, plus the new error η. Unrolling over the layers bounds the output error by $\mathcal{A}_L\,\eta$, where the *error-amplification factor* $\mathcal{A}_L$ depends only on K_ℓ and on norm bounds for the input, gates, and readout. Its explicit form is Equation 5 in Appendix D.2.

Theorem 3 (Approximate master theorem)**.** *Consider a gated linear program of the form in Equation 1 whose training examples include clean labels and possibly corrupted gate observations, and let $X \sim \mathcal{D}$ denote a fresh input. Suppose that, for some known positive constants R_0, G_ℓ, and Θ, we have $\|X\|_2 \le R_0$ and $\|s^{(\ell)}(X)\|_2, \|\widehat{s}^{(\ell)}(X)\|_2 \le G_\ell$ for every $\ell \in [L]$, almost surely, and $\|\beta^\star\|_2 \le \Theta$. Suppose that, for any requested accuracy $\eta > 0$ and confidence $\rho > 0$, each layer admits a local learner that, trained on fresh reconstructed features after the earlier layers have been learned, satisfies Equation 4 with probability at least $1-\rho$; and that a readout learner, trained on a fresh batch after all gates have been learned, returns $\widehat{\beta}$ whose L_2 error on $\widehat{\Phi}^{(L)}$ is within η of the best linear readout of norm at most Θ, with probability at least $1-\rho$.*

Then, for every $\varepsilon, \delta \in (0,1)$, running all learners with $\eta = \varepsilon/(2\max\{1, \mathcal{A}_L\})$ and $\rho = \delta/(2L)$ yields an improper predictor $\widehat{f}(x) = \langle \widehat{\beta}, \widehat{\Phi}^{(L)}(x) \rangle$ that uses only x at test time and satisfies

$$\|\widehat{f} - f^\star\|_{L_2} \le \varepsilon$$

with probability at least $1-\delta$. If the path features are computable in polynomial time, the sample complexity and running time are polynomial in L, the path-feature dimensions, $1 + \mathcal{A}_L$, $1/\varepsilon$, and $\log(1/\delta)$, together with the costs of the local and readout learners.

The proof and sample accounting are given in Appendix D.2.

Equation 4 holds, for example, for gates that are linear in the preceding hidden state, $s^{(\ell)} = V_\ell h^{(\ell-1)}$, as in a gated linear unit without the sigmoid. The network is still nonlinear, since each gate multiplies its branch and the output is a polynomial of degree at most 2^L in the input. Suppose the recorded gates carry bounded, mean-zero noise, as with stochastic rounding. Least squares then targets the clean gates, and linearity makes the gate error grow at most linearly in the feature error.

Corollary 4 (Noisy linear gates)**.** *Consider a gated linear program satisfying the bounds of Theorem 3, and suppose each gate is linear in the preceding path feature, $s^{(\ell)}(X) = B_\ell^\star \Phi^{(\ell-1)}(X)$ with $\|B_\ell^\star\|_{\mathrm{F}} \le \Lambda_\ell$. The observed gates are $\widetilde{s}^{(\ell)} = s^{(\ell)} + \xi_\ell$, where $\mathbb{E}[\xi_\ell \mid X] = 0$ and $\|\xi_\ell\|_2 \le \sigma_\ell$ almost surely, with known bounds $\Lambda_\ell, \sigma_\ell$. Then, for every $\varepsilon, \delta \in (0,1)$, there is an improper learner that uses only the input at test time and, with probability at least $1-\delta$, returns a predictor satisfying $\|\widehat{f} - f^\star\|_{L_2} \le \varepsilon$. Its sample complexity and arithmetic running time are polynomial in L, the path dimensions, R_0, G_ℓ, Λ_ℓ, σ_ℓ, Θ, $R_{\max} = \max_{0 \le r \le L} R_0 \prod_{t=1}^{r} G_t$, $1 + \mathcal{A}_L$, $1/\varepsilon$, and $\log(1/\delta)$, where $\mathcal{A}_L$ is given by Equation 5 with $K_\ell = \Lambda_\ell$.*

The proof is given in Appendix D.6. The factor $\mathcal{A}_L$ measures how much local gate errors can grow before reaching the output. If each layer doubles an incoming error, $\mathcal{A}_L$ can grow like 2^L, so the guarantee is polynomial in depth only when this amplification is controlled.

4 Computational limitations and a tractable margin regime

Revealing the gates need not reveal the numerical state carried by the computation. The following theorem shows that this gap is real: for both ReLU activation signs and hardmax attention routes, no learner is polynomial jointly in all architectural parameters in general, even when the trace is available at prediction time.

Theorem 4 (Hardness with revealed discrete traces)**.** *Under the cryptographic assumptions of (Kearns & Valiant, 1994), neither of the following classes is distribution-free PAC learnable in polynomial time, even when the learner receives the complete discrete trace during training and prediction and produces a polynomial-time evaluable hypothesis:*

1. *Bias-free ReLU networks of input dimension d, width at most n, and L hidden layers with a linear readout;*

2. *Residual hardmax-attention networks of token dimension d, sequence length N, and depth L, with a linear readout.*

Here polynomial time is measured in the stated model parameters, $1/\varepsilon$, and $\log(1/\delta)$; the loss is output disagreement.

Full proof can be found in Appendix F.1.

Why the traces do not remove the hardness. In both reductions, we construct the network so that its trace reveals which branches are active but not what those branches compute. We start from a function class that is hard to learn from labeled examples, namely polynomial-size acyclic deterministic finite automata (Kearns & Valiant, 1994; Hopcroft et al., 2001), which read an input bit by bit while updating a hidden state. We simulate each such target by a network of polynomial size whose routing depends only on the input: at step t the active branch is determined by the input bit x_t alone, while the hidden state, and hence the target, is carried by the linear maps inside the branches. The trace is therefore the same function of x for every target, and a learner can compute it itself during training and prediction. Traced examples are thus no more informative than ordinary labeled ones, so an efficient learner from traces would efficiently learn the hard class.The ReLU and hardmax-attention constructions differ. Full details are given in Appendices F.2 and F.3.

These results rule out a general polynomial-time learner from discrete traces alone. They leave open whether additional structure on the local prediction problems can make learning efficient. We next show that learning becomes efficient for ReLU networks under a margin assumption on the gates and a norm bound on the readout.

4.1 Polynomial-time learning under a path-margin assumption

For a bias-free ReLU network with widths $n_1, \dots, n_L$, Proposition 1 shows that each hidden gate $s_j^{(\ell)}$ is a halfspace in $\Phi^{(\ell-1)}$ and the output is linear in $\Phi^{(L)}$. These path features have dimension $d\prod_{r<\ell} n_r$, which can be exponential in depth.

However, the feature need never be formed, because inner products of tensor products factor:

$$K_{\ell-1}(x,x') = \langle \Phi^{(\ell-1)}(x), \Phi^{(\ell-1)}(x')\rangle = \langle x, x'\rangle \prod_{r<\ell} \langle s^{(r)}(x), s^{(r)}(x')\rangle,$$

which costs only $O(d + \sum_{r<\ell} n_r)$. We learn each gate with a kernel perceptron, which never needs $\Phi^{(\ell-1)}$ explicitly. Its predictions are weighted sums of kernel evaluations on its past mistakes. Norm-constrained kernel least squares gives a readout of the form $\sum_i \alpha_i K_L(x_i, x)$, clipped to $[-B, B]$. Neither predictor constructs the exponential-size path features.

Cheap evaluations alone do not make the learner efficient: a perceptron might make too many mistakes, and a kernel regressor might need too many samples. We therefore assume a *path margin* γ for every hidden gate and a norm bound B on the readout (Appendix G gives the precise statement). Then each perceptron makes $O(1/\gamma^2)$ mistakes, and the readout needs a number of samples polynomial in B, so the whole procedure is polynomial in the network size, $1/\gamma$, and B.

Theorem 5 (Efficient learning from sign traces under a path margin)**.** *Consider a bias-free ReLU network with widths $n_1, \dots, n_L$ and real-valued linear readout $f^\star(x) = c^\top h^{(L)}(x)$; by Proposition 1, $f^\star(x) = \langle \beta, \Phi^{(L)}(x)\rangle$ with $\beta = B_L^\top c$. Let $M = \sum_{\ell=1}^{L} n_\ell$ be the number of hidden gates. Fix $\varepsilon, \delta \in (0,1)$. Assume:*

1. Path margin *with known $\gamma \in (0,1]$: for every hidden gate (ℓ, j), conditional on $\Phi^{(\ell-1)}(X) \neq 0$, either $s_j^{(\ell)}(X)$ is almost surely constant, or there is a unit vector $u_{\ell,j}$ such that, almost surely, $\big(2s_j^{(\ell)}(X) - 1\big)\big\langle u_{\ell,j},\, \Phi^{(\ell-1)}(X)/\|\Phi^{(\ell-1)}(X)\|\big\rangle \geq \gamma$.*

2. Norm bound*: $\|\beta\| \leq \Theta$ and $\Theta\,\|\Phi^{(L)}(X)\| \leq B$ almost surely, with Θ and B known.*

Then a learner given ordered hidden activation signs and labels during training returns a predictor $\widehat{f}$ that uses only x at test time and satisfies $\|\widehat{f} - f^\star\|_{L_2} \leq \varepsilon$ with probability at least $1-\delta$. Its sample complexity and arithmetic running time are polynomial in d, M, B, $1/\gamma$, $1/\varepsilon$, and $\log(1/\delta)$.

At test time, the learner reconstructs the signs layer by layer. Until the first incorrect sign, each gate receives its true prefix, so the readout sees the true feature except with probability at most the sum of the local gate errors. Appendix G.1 gives the precise treatment of zero features and constant gates, the sample bound, and the proof.

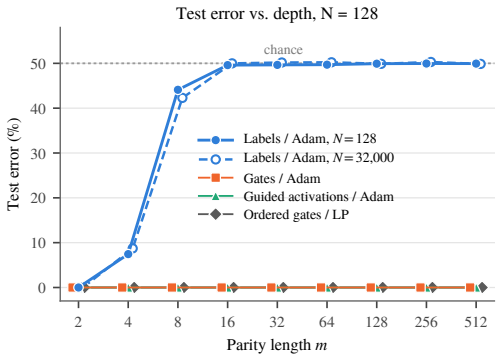
5 Experiments on Parity

We evaluate whether intermediate supervision helps neural networks learn the parity function, a standard example for which output-only gradient-based training is challenging. The experiment uses a hard-coded teacher, so the target computation and all intermediate signals are known exactly.

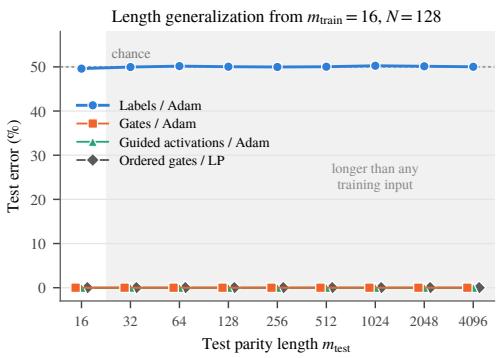
Task, teacher, and students. For a power-of-two length m, inputs are sampled uniformly from $\{0,1\}^m$ and the target is $y = f^\star(x) = x_1 \oplus \cdots \oplus x_m$. The teacher is a balanced, bias-free ReLU tree based on the identity $\mathrm{XOR}(u,v) = \mathrm{ReLU}(u-v) + \mathrm{ReLU}(v-u)$, with $u, v \in \{0,1\}$. Thus, the teacher has depth $L = \log_2 m$ and $2(m-1)$ ReLU units. Each neural student uses the same tree and shares one trainable 2×2 matrix across all nodes and layers. This tied-weight design permits evaluation at depths not observed during training. The teacher weights are never given to the student.

Supervision and protocol. We compare output-label supervision, binary-gate supervision, guided activation supervision, and the constructive ordered-gate LP learner. All methods receive only x at test time. We use five seeds and $N = 128$ training examples at lengths $m \in \{2, 4, \ldots, 512\}$; labels-only training is also run with $N = 32{,}000$. The length-generalization experiment trains at $m_{\mathrm{train}} = 16$ and evaluates, without retraining, up to $m_{\mathrm{test}} = 4096$. Each reported test error uses 20,000 independent examples. Appendix H provides the losses and other experimental details.

Results. Figure 1a varies the parity length m with $N = 128$ training examples, and includes a labels-only experiment with $N = 32{,}000$. Labels-only training fits only the shallowest trees ($m \leq 4$) and performs near chance from $m = 16$ onward, even with $128\times$ more examples. All three intermediate-supervision methods, however, attain zero error at every tested depth. These methods not only solve the problem but also learn the teacher's local structure, enabling strong generalization: in Figure 1b, students trained at $m = 16$ maintain zero error up to $m = 4096$, $256\times$ longer than any training input. Intermediate supervision thus identifies a reusable local XOR operation that composes far beyond the training depth.



(a) Test error versus parity length m at $N = 128$; the dashed line is labels-only with $N = 32{,}000$.



(b) Length generalization after training at $m_{\mathrm{train}} = 16$ with $N = 128$.

6 Conclusion and Future Work

We formulated intermediate supervision as PAC learning of gated linear programs with training-time access to selected internal computations. Our results show that the information revealed can fundamentally change the computational tractability of learning. Complete numerical traces enable polynomial-time learning given learnable gates, while partial traces admit exact composition guarantees but face cryptographic lower bounds; stability bounds address inexact observations, and parity experiments demonstrate successful learning where the labels-only gradient-based baseline fails. The main open question is which structural conditions, beyond path margins, make discrete traces efficient at every depth; answering it would indicate which parts of a computation are worth revealing when training deep models.

REPRODUCIBILITY STATEMENT

All theoretical results are stated with their full assumptions, and complete proofs are given in the appendices: Appendix B covers complete traces; Appendix C covers path representations, the exact master theorem, local exact-recovery oracles, and residual attention; Appendix D covers inexact traces and noisy linear gates; Appendix E covers learning from a fixed unknown gate order; Appendix F gives the hardness reductions; and Appendix G proves the path-margin result. For the parity experiments, Appendix H specifies the teacher, all training objectives, optimization hyperparameters, restarts, random seeds (including the master seed), and the evaluation protocol, and reports complete numerical results.

ETHICS STATEMENT

This paper is primarily theoretical: it studies how intermediate supervision affects the computational complexity of PAC learning deep models. Our experiments use only synthetic parity data generated by a hard-coded teacher. The work involves no human-subject data, no personal or sensitive information, and no direct high-risk application.

AI USE STATEMENT

All research ideas, results, and proofs in this paper were developed by the authors. AI tools were used only to assist with writing, editing, and organizing the text, and to help refine the formulation of some mathematical statements. All mathematical statements and proofs were independently checked by the authors, and all cited literature was verified against the original sources. The authors take full responsibility for the correctness of the results, the accuracy of the citations, the novelty claims, and the final content of the paper.

REFERENCES

Yaser Abu-Mostafa. A method for learning from hints. *Advances in Neural Information Processing Systems*, 5, 1992.

Yaser S Abu-Mostafa. Learning from hints in neural networks. *Journal of complexity*, 6(2):192–198, 1990.

Yaser S Abu-Mostafa. Hints and the vc dimension. *Neural Computation*, 5(2):278–288, 1993.

Yaser S Abu-Mostafa. Hints. *Neural computation*, 7(4):639–671, 1995.

Awni Altabaa, Omar Montasser, and John Lafferty. Cot information: Improved sample complexity under chain-of-thought supervision. *Advances in Neural Information Processing Systems*, 38: 24822–24862, 2025.

Peter L Bartlett, Nick Harvey, Christopher Liaw, and Abbas Mehrabian. Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks. *Journal of Machine Learning Research*, 20(63):1–17, 2019.

Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Learnability and the vapnik-chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4):929–965, 1989.

Sitan Chen and Yuanzhi Li. Provably learning a multi-head attention layer. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pp. 1744–1754, 2025.

Sitan Chen, Adam R Klivans, and Raghu Meka. Learning deep relu networks is fixed-parameter tractable. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 696–707. IEEE, 2021.

Sitan Chen, Aravind Gollakota, Adam Klivans, and Raghu Meka. Hardness of noise-free learning for two-hidden-layer neural networks. *Advances in Neural Information Processing Systems*, 35: 10709–10724, 2022.

Sitan Chen, Zehao Dou, Surbhi Goel, Adam Klivans, and Raghu Meka. Learning narrow one-hidden-layer relu networks. In *The Thirty Sixth Annual Conference on Learning Theory*, pp. 5580–5614. PMLR, 2023.

Amit Daniely and Gal Vardi. From local pseudorandom generators to hardness of learning. In *Conference on Learning Theory*, pp. 1358–1394. PMLR, 2021.

Ilias Diakonikolas, Daniel M Kane, Vasilis Kontonis, and Nikos Zarifis. Algorithms and sq lower bounds for pac learning one-hidden-layer relu networks. In *Conference on Learning Theory*, pp. 1514–1539. PMLR, 2020.

Ilan Doron-Arad, Idan Mehalel, and Elchanan Mossel. Stochastic autoregressive learning. *arXiv preprint arXiv:2608.07224*, 2026.

Benjamin L Edelman, Surbhi Goel, Sham Kakade, and Cyril Zhang. Inductive biases and variable creation in self-attention mechanisms. In *International Conference on Machine Learning*, pp. 5793–5831. PMLR, 2022.

Panagiotis Giannoulis, Yorgos Pantis, and Christos Tzamos. Teaching transformers to solve combinatorial problems through efficient trial & error. In *Proceedings of the Thirtyninth Annual Conference on Neural Information Processing Systems*, volume 38, pp. 133548–133580, 2025.

Steve Hanneke, Idan Mehalel, and Shay Moran. Sample complexity of autoregressive reasoning: Chain-of-thought vs. end-to-end. *arXiv preprint arXiv:2604.12013*, 2026.

Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 3779–3787, 2019.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

John E Hopcroft, Rajeev Motwani, and Jeffrey D Ullman. Introduction to automata theory, languages, and computation. *Acm Sigact News*, 32(1):60–65, 2001.

Nirmit Joshi, Gal Vardi, Adam Block, Surbhi Goel, Zhiyuan Li, Theodor Misiakiewicz, and Nathan Srebro. A theory of learning with autoregressive chain of thought. *arXiv preprint arXiv:2503.07932*, 2025.

Michael Kearns and Leslie Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *Journal of the ACM*, 41(1):67–95, 1994.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*, pp. 5338–5348. Pmlr, 2020.

Animesh Koratana, Daniel Kang, Peter Bailis, and Matei Zaharia. Lit: Learned intermediate representation training for model compression. In *International Conference on Machine Learning*, pp. 3509–3518. PMLR, 2019.

Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Artificial intelligence and statistics*, pp. 562–570. Pmlr, 2015.

Shuchen Li, Ilias Zadik, and Manolis Zampetakis. On the hardness of learning one hidden layer neural networks. *arXiv preprint arXiv:2410.03477*, 2024.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*, 2021.

Dmitry Pechyony and Vladimir Vapnik. On the theory of learning with privileged information. *Advances in neural information processing systems*, 23, 2010.

Du Phan, Matthew Douglas Hoffman, David Dohan, Sholto Douglas, Tuan Anh Le, Aaron Parisi, Pavel Sountsov, Charles Sutton, Sharad Vikram, and Rif A Saurous. Training chain-of-thought via latent-variable inference. *Advances in Neural Information Processing Systems*, 36:72819–72841, 2023.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *Proceedings of ICLR*, 2015.

Michal Sharoni and Sivan Sabato. On the capacity limits of privileged erm. In *International Conference on Artificial Intelligence and Statistics*, pp. 523–534. PMLR, 2023.

Jacob Trauger and Ambuj Tewari. Sequence length independent norm-based generalization bounds for transformers. In *International Conference on Artificial Intelligence and Statistics*, pp. 1405–1413. PMLR, 2024.

Leslie G Valiant. A theory of the learnable. In *Foundations of Computation and Machine Learning: The Work of Leslie Valiant*, pp. 69–94. 2026.

Vladimir Vapnik and Rauf Izmailov. Learning using privileged information: similarity control and knowledge transfer. *The Journal of Machine Learning Research*, 16(1):2023–2049, 2015.

Vladimir Vapnik and Akshay Vashist. A new learning paradigm: Learning using privileged information. *Neural networks*, 22(5-6):544–557, 2009.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Zixuan Wang, Eshaan Nichani, Alberto Bietti, Alex Damian, Daniel Hsu, Jason D Lee, and Denny Wu. Learning compositional functions with transformers from easy-to-hard data. *arXiv preprint arXiv:2505.23683*, 2025.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 2022.

Chenxiao Yang, Nathan Srebro, and Zhiyuan Li. Tight sample complexity of transformers. *arXiv preprint arXiv:2606.09731*, 2026.

APPENDIX

A RELATED WORK

Intermediate supervision, hints, and privileged information. The idea that a learner may benefit from information beyond input–output pairs predates modern deep learning. Abu-Mostafa introduced *learning from hints*, where additional properties of the target function are supplied to the learner, and developed a general framework for incorporating heterogeneous hints into learning (Abu-Mostafa, 1990; 1992; 1993; 1995). Learning Using Privileged Information (LUPI) similarly augments the training examples with information that is unavailable at prediction time (Vapnik & Vashist, 2009; Pechyony & Vapnik, 2010; Vapnik & Izmailov, 2015). Recent work has clarified that privileged information does not automatically improve worst-case statistical guarantees: for example, (Sharoni & Sabato, 2023) show that the capacity of Privileged ERM can be no smaller than that of ordinary ERM unless the privileged component is itself suitably controlled.

Related ideas appear throughout deep learning. Deeply supervised networks attach auxiliary objectives to intermediate layers (Lee et al., 2015); concept bottleneck models first predict human-provided concepts and then use them for the final prediction (Koh et al., 2020); and knowledge distillation augments endpoint labels with information produced by a teacher (Hinton et al., 2015).

Distillation methods have progressively moved from teacher outputs to internal representations: FitNets match hidden representations (Romero et al., 2015), learned intermediate representation training transfers intermediate teacher blocks (Koratana et al., 2019), and (Heo et al., 2019) supervise the activation boundaries of hidden neurons, which is particularly close to the ReLU gates considered here.

Our setting shares the training-only nature of privileged information but exploits additional structure: the revealed variables are pieces of the target model's own computation. We ask when these variables can be reconstructed from the input and when, after revealing them, the remaining hidden computation belongs to a tractable learning problem. Accordingly, our focus is not only whether intermediate information improves generalization or optimization, but whether it changes *computational PAC learnability*. We give explicit polynomial-time learners in positive regimes and complementary hardness results showing that even complete discrete traces need not make learning efficient.

Chain-of-thought and process supervision. The recent success of reasoning models has renewed interest in supervising intermediate computation directly. Empirically, process supervision provides feedback on individual reasoning steps rather than only on the final outcome; (Lightman et al., 2024), for example, report substantial gains from step-level supervision on mathematical reasoning.

A recent theoretical literature studies the statistical and computational consequences of CoT supervision. Most closely related to our question, (Joshi et al., 2025) formulate a PAC model in which a time-invariant next-token generator is iterated to produce an autoregressive chain. They compare learning when the full chain is observed with learning from the final answer alone, analyze both sample and computational complexity, and give settings in which CoT learning is computationally tractable. (Altabaa et al., 2025) introduce a CoT-information quantity that characterizes the statistical advantage provided by observing a reasoning trace. (Hanneke et al., 2026) further characterize the dependence of sample complexity on reasoning length, showing that full CoT supervision can remove a dependence on the number of autoregressive steps that is present under endpoint supervision. Recent work also extends these questions to stochastic autoregressive generators (Doron-Arad et al., 2026).

Our work studies a complementary form of intermediate supervision. Rather than observing the sequence generated by an autoregressive next-token model, we reveal selected variables in the *internal computation* of a target neural architecture, such as ReLU activation gates or attention-routing decisions. This distinction creates a different computational issue: the observed gates may reveal the control flow of a computation while leaving its linear content hidden. Our positive results identify conditions under which this hidden content admits an efficiently learnable representation, whereas our lower bounds show that even a complete activation or routing trace can leave the underlying learning problem computationally hard.

Computational learning of neural networks. For neural networks, finite statistical capacity does not imply computationally efficient learnability. VC- and pseudodimension bounds for piecewise-linear networks give polynomial statistical complexity (Bartlett et al., 2019), while efficient algorithms typically require additional assumptions on the distribution, architecture, width, or parameters. (Diakonikolas et al., 2020) study one-hidden-layer ReLU networks under Gaussian marginals, giving efficient algorithms for positive output weights and statistical-query lower bounds for general coefficients; (Chen et al., 2021) give fixed-parameter algorithms for deep ReLU networks under Gaussian inputs; and (Chen et al., 2023) study narrow one-hidden-layer networks. Complementary hardness results show that even simple ReLU classes can remain computationally intractable: (Daniely & Vardi, 2021) derive cryptographic hardness from local pseudorandom generators, (Chen et al., 2022) prove statistical-query and cryptographic hardness for noise-free two-hidden-layer ReLU networks, including under Gaussian inputs, and (Li et al., 2024) obtain hardness for polynomial-size one-hidden-layer networks under standard Gaussian inputs with polynomially small additive Gaussian label noise.

Our results vary a different axis: the information available during training. With ordered activation gates, we obtain distribution-free learning guarantees with polynomial sample and running time at every fixed depth. Conversely, even the complete activation-sign trace does not yield a learner

polynomial jointly in dimension, width, depth, and inverse accuracy in the general distribution-free setting.

Learning attention and transformers. Recent work studies both the statistical and computational complexity of attention-based models. (Edelman et al., 2022) give sample-complexity guarantees for sparse functions with weak dependence on sequence length, (Trauger & Tewari, 2024) derive norm-based generalization bounds without explicit sequence-length dependence, and (Yang et al., 2026) obtain nearly matching VC-dimension bounds and corresponding sample-complexity results, including for teacher-forced chain-of-thought learning. On the algorithmic side, (Chen & Li, 2025) give a provable algorithm for learning a multi-head softmax-attention layer under non-degeneracy assumptions, with polynomial complexity for a constant number of heads and a matching worst-case exponential dependence on the number of heads. (Wang et al., 2025) show that a compositional task representable by a logarithmic-depth Transformer has a statistical-query barrier under endpoint-only learning, while suitable easy-to-hard curricula permit efficient learning by gradient descent.

Our attention results instead use routing information revealed during training. We study residual attention with hardmax routes or softmax attention rows, which reduces routing recovery to local linear or linearized prediction problems and enables test-time reconstruction. The resulting guarantees are distribution-free under the stated trace assumptions, while our hardmax lower bound shows that even complete routing information need not eliminate computational hardness.The attention model analyzed here is a residual, single-head self-attention sublayer without layer normalization.

B FULL TRACES: PROOFS

Throughout this section the training record contains, for every training input, the gate vectors $s^{(1:L)}$, the states $h^{(1:L)}$, and the label. Recall $d_\ell = \dim h^{(\ell)}$, $d_0 = d$, and $\bar{A}_\ell = [A_{\ell,1} \ \cdots \ A_{\ell,k_\ell}]$, so that $h^{(\ell)} = \bar{A}_\ell(s^{(\ell)} \otimes h^{(\ell-1)})$.

B.1 PROOF OF THEOREM 1

Theorem (Theorem 1: Complete traces for gated linear programs)**.** *Consider a gated linear program of the form in Equation 1. Suppose that, for each training input, the teacher reveals $s^{(1)}, h^{(1)}, \ldots, s^{(L)}, h^{(L)}$ and the final label. Assume that, for every layer ℓ, the gate rule $s^{(\ell)} = g_\ell(h^{(\ell-1)})$ is PAC learnable under disagreement loss, with training and prediction costs uniformly polynomial in the parameters below.*

Then, for any $\varepsilon, \delta \in (0,1)$, there exists an improper learner that uses only the input at test time and, with probability at least $1-\delta$, satisfies

$$\Pr_x[\widehat{f}(x) \neq f^\star(x)] \leq \varepsilon.$$

Its sample complexity and running time are polynomial in L, $\max_\ell d_\ell$, $\max_\ell k_\ell d_{\ell-1}$, $1/\varepsilon$, and $\log(1/\delta)$.

Proof. The revealed states let us learn each layer on its true inputs. We first describe these local fits, then show that their guarantees remain sufficient when the learner uses its own predictions at test time.

Write $h^{(0)}(x) = x$, set $p_\ell = k_\ell d_{\ell-1}$, and define

$$V_\ell(x) = s^{(\ell)}(x) \otimes h^{(\ell-1)}(x) \in \mathbb{R}^{p_\ell}.$$

The layer equation is $h^{(\ell)}(x) = \bar{A}_\ell V_\ell(x)$, where $\bar{A}_\ell = [A_{\ell,1} \ \cdots \ A_{\ell,k_\ell}]$. Choose

$$\alpha = \frac{\varepsilon}{2L+1}, \qquad \rho = \frac{\delta}{2L+1}.$$

There are L gate fits, L state fits, and one readout fit; each will have disagreement probability at most α, with failure probability at most ρ.

Learning from the recorded values. For each layer ℓ, use the assumed gate learner on the observed pairs $(h^{(\ell-1)}(x_i), s^{(\ell)}(x_i))$ to obtain $\widehat{g}_\ell$. If $m_\ell(\alpha, \rho)$ denotes its sample bound, then with probability at least $1 - \rho$,

$$\Pr_X[\widehat{g}_\ell(h^{(\ell-1)}(X)) \neq s^{(\ell)}(X)] \leq \alpha.$$

Here disagreement means that the whole gate vector is not recovered.

Next, solve the linear equations

$$\widehat{\bar{A}}_\ell V_\ell(x_i) = h^{(\ell)}(x_i)$$

for $\widehat{\bar{A}}_\ell$. They are feasible because the teacher's matrix satisfies them. Every solution agrees with $\bar{A}_\ell$ on the span of the observed vectors $V_\ell(x_i)$. By Lemma 1,

$$m \geq \frac{2}{\alpha}\left(p_\ell \log 2 + \log \frac{1}{\rho}\right)$$

ensures, with probability at least $1 - \rho$, that

$$\Pr_X[\widehat{\bar{A}}_\ell V_\ell(X) \neq h^{(\ell)}(X)] \leq \alpha.$$

The same span event controls all d_ℓ output coordinates at once. The data need not span $\mathbb{R}^{p_\ell}$, and the fitted matrix need not equal the teacher's matrix away from the observed span.

Finally, fit $\widehat{c}$ by solving

$$\langle \widehat{c}, h^{(L)}(x_i)\rangle = f^\star(x_i).$$

The teacher's readout c is feasible. Applying the same lemma in dimension d_L gives

$$\Pr_X[\langle \widehat{c}, h^{(L)}(X)\rangle \neq f^\star(X)] \leq \alpha$$

with probability at least $1 - \rho$, using $2(d_L \log 2 + \log(1/\rho))/\alpha$ examples.

All these fits may share one batch of complete traces: each local sample still consists of independent examples from its required distribution. A gate learner can use the first $m_\ell(\alpha, \rho)$ examples if it needs fewer than the full batch. A sufficient total sample size is

$$m = \left\lceil \max\left\{ \max_{\ell \in [L]} m_\ell(\alpha, \rho),\ \frac{2}{\alpha}\left(\max\{d_L, p_1, \ldots, p_L\} \log 2 + \log \frac{1}{\rho}\right)\right\}\right\rceil.$$

A union bound over the $2L + 1$ fits shows that all their disagreement bounds hold together with probability at least $1 - (2L + 1)\rho = 1 - \delta$. Independence between the fits is not needed.

Prediction from the input alone. For a new input x, compute

$$\widehat{h}^{(0)}(x) = x, \qquad \widehat{s}^{(\ell)}(x) = \widehat{g}_\ell(\widehat{h}^{(\ell-1)}(x)), \qquad \widehat{h}^{(\ell)}(x) = \widehat{\bar{A}}_\ell\big(\widehat{s}^{(\ell)}(x) \otimes \widehat{h}^{(\ell-1)}(x)\big),$$

and return $\widehat{f}(x) = \langle \widehat{c}, \widehat{h}^{(L)}(x)\rangle$. As required for any such sequential predictor, the learned gate rules are defined on all possible input states, including incorrect ones.

Fix a training outcome for which all the local bounds hold. Let B_ℓ^g be the set where the gate fit errs on the *true* state, let B_ℓ^h be the set where the state fit errs on the *true* vector $V_\ell(x)$, and let B^y be the set where the readout errs on the *true* final state. Each set has probability at most α. If x lies outside their union, induction gives $\widehat{h}^{(\ell)}(x) = h^{(\ell)}(x)$ for every layer. Indeed, the preceding state is initially correct. Whenever it is correct at layer $\ell - 1$, the gate learner receives its true input; since $x \notin B_\ell^g$, it returns the true gate vector. The state fit then receives $V_\ell(x)$ and, since $x \notin B_\ell^h$, returns the true next state. At the last layer, $x \notin B^y$ also makes the readout correct. Therefore

$$\Pr_X[\widehat{f}(X) \neq f^\star(X)] \leq \sum_{\ell=1}^{L}\big(\Pr_X[B_\ell^g] + \Pr_X[B_\ell^h]\big) + \Pr_X[B^y] \leq (2L + 1)\alpha = \varepsilon.$$

We do not need to control what happens after a first error: every incorrect prediction is already covered by this union.

Computational cost. The covariate for a state fit has dimension $p_\ell = k_\ell d_{\ell-1}$; the readout covariate has dimension d_L. Forming all V_ℓ on the training batch costs $O(m \sum_\ell p_\ell)$ arithmetic operations. Each state fit is a linear system with p_ℓ unknowns per output coordinate, and the readout is a linear system with d_L unknowns. These systems can be solved in time polynomial in m, p_ℓ, and d_ℓ. At prediction time, beyond evaluating the learned gates, forming the covariates and multiplying by the fitted matrices costs $O(\sum_\ell (p_\ell + d_\ell p_\ell) + d_L)$ operations. Together with the assumed polynomial gate-learning costs and the displayed choice of m, this proves the stated bounds. In particular, no feature dimension contains a product of the widths over layers. Under the paper's arithmetic convention these are arithmetic costs; for rational data, solving the linear systems also has polynomial bit cost in the encoding length of the recorded values. The costs of evaluating and encoding gate predictions are part of the assumed gate-learning routines. □

B.2 Proof of Corollary 1

Corollary (Corollary 1: ReLU networks with complete activation values)**.** *A depth-L, width-at-most-n ReLU network with a linear readout is distribution-free PAC learnable from exact hidden activation values and labels. For every $\varepsilon, \delta \in (0,1)$, there is a proper learner that uses only the input at test time and, with probability at least $1-\delta$, satisfies*

$$\Pr_x[\widehat{f}(x) \neq f^\star(x)] \leq \varepsilon.$$

Its sample complexity and running time are polynomial in d, n, L, $1/\varepsilon$, and $\log(1/\delta)$.

Proof. Pad every layer to width exactly n with units that are identically zero, so $d_\ell \leq n$ for $\ell \geq 1$ and $d_0 = d$. The gate proper of a ReLU unit is its activation bit, but the state determines the bit and is itself a numerical ReLU gate on the preceding state,

$$h_j^{(\ell)} = \sigma\Big(\langle w_j^{(\ell)}, h^{(\ell-1)} \rangle\Big), \qquad j \in [n],\ \ell \in [L],$$

so we can learn the numerical activations directly: take the $M = nL$ scalar blocks $h_j^{(\ell)}$ in layer order, with U selecting $h^{(\ell-1)}$ and F selecting $h^{(L)}$. These only select previously revealed states and are therefore admissible. Moreover, $f^\star = \langle c, h^{(L)} \rangle$ with $D = d_L \leq n$.

Each block is a numerical ReLU gate on a covariate of dimension $p_\ell = d_{\ell-1} \leq \max\{d, n\} \leq d + n$, so case (iii) of Corollary C.4 supplies an exact-recovery learner from $O((p_\ell \log(2/\alpha) + \log(2/\rho))/\alpha)$ samples. Its algorithm is the linear program

$$\langle \widehat{w}, h_i^{(\ell-1)} \rangle = h_{j,i}^{(\ell)} \quad \text{when } h_{j,i}^{(\ell)} > 0, \qquad \langle \widehat{w}, h_i^{(\ell-1)} \rangle \leq 0 \quad \text{when } h_{j,i}^{(\ell)} = 0,$$

which is feasible because $w_j^{(\ell)}$ satisfies it, including when the true preactivation is exactly zero, and which is of size polynomial in the sample and in p_ℓ. For a padded unit every constraint is an inequality and $\widehat{w} = 0$ is feasible, so realizability holds and case (iii) applies; the solver is not required to return $\widehat{w} = 0$, and its guarantee does not depend on which feasible point it returns.

Substituting $\alpha = \varepsilon/(2nL)$ and $\rho = \delta/(2nL)$ into Theorem 2 gives a shared gate batch of

$$O\left(\frac{nL}{\varepsilon}\left((d+n)\log\frac{4nL}{\varepsilon} + \log\frac{4nL}{\delta}\right)\right)$$

recorded traces. The readout batch is $O((n + \log(2/\delta))/\varepsilon)$ and the display absorbs it term by term: $\frac{nL}{\varepsilon}(d+n)\log\frac{4nL}{\varepsilon} \geq \frac{n}{\varepsilon}\log 4$ absorbs n/ε, and $\frac{nL}{\varepsilon}\log\frac{4nL}{\delta} \geq \frac{1}{\varepsilon}\log\frac{2}{\delta}$ absorbs $\log(2/\delta)/\varepsilon$ because $nL \geq 1$ and $4nL/\delta > 2/\delta$. All n blocks of a layer share one batch, which the proof of Theorem 2 permits, since sharing an example across gates does not change any marginal.

For a biased network, $h_j^{(\ell)} = \sigma(\langle w_j^{(\ell)}, h^{(\ell-1)} \rangle + b_j^{(\ell)})$, append a constant coordinate to the local covariate and fit (w, b) jointly; the local dimension becomes $d_{\ell-1} + 1$ and the readout is linear in $(h^{(L)}, 1)$. This needs no change to the recorded trace, unlike the state augmentation of Appendix C.1, which is needed only when the state itself has to stay affine. On rational data the linear programs and the linear system are solved in time polynomial in the bit length of the input, which includes the encoding length of the recorded activation values. No norm bound on $W^{(\ell)}$ or c is used anywhere. □

Exactness is a hypothesis, not a convenience. On a rounded record the exact guarantee of Corollary 1 need not apply. The layer-ℓ program imposes $\langle \widehat{w}, h_i^{(\ell-1)} \rangle = h_{j,i}^{(\ell)}$ on every example with a positive recorded value, and $w_j^{(\ell)}$ satisfies those equalities only because the record is the number the teacher computed. Round the record and the true vector is no longer feasible, and the constraints can become inconsistent. Take $d = n = L = 1$, $w^\star = 1/3$, and x uniform on $\{1, 2\}$: the values $1/3$ and $2/3$, rounded to two places, demand $\widehat{w} = 0.33$ and $2\widehat{w} = 0.67$ at once, so as soon as the sample contains both inputs the program is infeasible. Rounding can also move an example across the boundary between the equality and inequality constraints, which is exactly what the consistency step of Corollary C.4(iii) uses. Neither failure is repaired downstream: Theorem 2 composes exact agreement; after a first erroneous value, later covariates need not satisfy its local guarantees. Rounded full traces require a separate robust local-learning contract that accounts for corrupted covariates as well as targets; Corollary 4 does not establish this extension.

Where the depth went. Corollary 2 learns the same networks from activation bits, and its local covariate is $\Phi^{(\ell-1)}$ of dimension $dn^{\ell-1}$, because the bits leave $h^{(\ell-1)}$ undetermined and Proposition 1 is what restores linearity. With the values recorded, $h^{(\ell-1)}$ is available at test time by construction, the local covariate is the state, and the product over layers never forms. The two bounds differ only in this dimension.

Consistency with Section 4. Corollary 1 does not contradict Theorem 4, and the reason is visible in that theorem's proof. The reduction there needs the trace to be computable from the input by a map that does not know the target, and it is: the activation pattern of the simulating network equals $\Gamma_{S,T}(x)$. The activation values are not so computable. The state block of that network is $\mathbf{1} + e_{q_t}$, which names the automaton's state after t steps, so a reduction that could produce the value trace would already know the automaton's run. This explains the distinction between complete numerical traces and complete sign traces.

C Path representations and exact-recovery proofs

All random functions below are measurable; the finite-dimensional classes admit the usual measurable choices of feasible solutions. Empty products equal one. Guarantees for local learners are evaluated against a new draw from the input distribution.

C.1 Proof of Proposition 1

Proposition (Proposition 1: Path linearization)**.** *Set $\Phi^{(0)} = x \in \mathbb{R}^d$ and, recursively, $\Phi^{(\ell)} = s^{(\ell)} \otimes \Phi^{(\ell-1)} \in \mathbb{R}^{D_\ell}$, where $u \otimes v = (u_1 v, \dots, u_k v)$ denotes the Kronecker product and $D_\ell = d \prod_{r=1}^{\ell} k_r$. Then, for every ℓ, there exists a fixed matrix $B_\ell \in \mathbb{R}^{d_\ell \times D_\ell}$ such that $h^{(\ell)} = B_\ell \Phi^{(\ell)}$. Consequently, there exists a fixed vector $\beta \in \mathbb{R}^{D_L}$ such that $f^\star(x) = \langle \beta, \Phi^{(L)} \rangle$.*

Proof. Set $B_0 = I_d$. If $h^{(\ell-1)} = B_{\ell-1}\Phi^{(\ell-1)}$, then

$$h^{(\ell)} = \sum_{j=1}^{k_\ell} s_j^{(\ell)} A_{\ell,j} B_{\ell-1} \Phi^{(\ell-1)} = \big[A_{\ell,1} B_{\ell-1} \ \cdots \ A_{\ell,k_\ell} B_{\ell-1} \big] \Phi^{(\ell)}.$$

This defines B_ℓ and proves the assertion by induction. Taking $\beta = B_L^\top c$ proves the readout claim. The dimension follows from concatenating k_ℓ copies of a $D_{\ell-1}$-dimensional vector. No sign, boundedness, or independence assumption on the gate coordinates is used.

For a ReLU network, let e_j be a coordinate vector and set $A_{\ell,j} = e_j e_j^\top W^{(\ell)}$. Then $s_j^{(\ell)} = a_j^{(\ell)}$ and the coordinate indexed by $(i, j_1, \dots, j_\ell)$ is $x_i \prod_{r=1}^{\ell} a_{j_r}^{(r)}$. This recovers the original path representation.

For affine content $\sum_j s_j(A_j h + b_j) + b_0$, augment the state to $\bar{h} = (h, 1)$. For each nonconstant gate use the matrix with upper blocks $[A_j \ b_j]$ and zero bottom row. Add an always-one gate whose matrix has upper blocks $[0 \ b_0]$ and bottom row $[0 \ 1]$. This preserves the constant coordinate at every

layer. An affine gate preactivation is also linear in $\bar{h}$. Merely appending one input coordinate without preserving it at subsequent layers is insufficient. The same construction includes residual maps in an always-one branch. For biased ReLU networks it suffices to use width $n+1$ and input dimension $d+1$ in the resulting bound. □

C.2 A SPAN LEMMA, INCLUDING BLOCKS OF EQUATIONS

Lemma 1 (Coverage by the span of independent observations)**.** *Let $\mathcal{V}(x)$ be a finite set of vectors in $\mathbb{R}^p$, constructed deterministically from a random input x. Draw independent $x_1, \dots, x_m$ and let V_m be the span of all vectors in their sets. For $\alpha \in (0,1)$,*

$$\Pr\Big[\Pr_x[\mathcal{V}(x) \not\subseteq V_m] > \alpha\Big] \le \exp(p \log 2 - m\alpha/2).$$

Consequently, if unknown $\theta^\star \in \mathbb{R}^p$ satisfies observed linear equations $\langle \theta^\star, v\rangle = b(x,v)$ for every $v \in \mathcal{V}(x)$, any interpolating $\widehat{\theta}$ satisfies all equations on a fresh input except with probability α, with confidence $1-\rho$, when $m \ge 2(p\log 2 + \log(1/\rho))/\alpha$.

Proof. Let $r_t = \dim V_t$, $q_t = \Pr_x[\mathcal{V}(x) \not\subseteq V_t]$, and $V_0 = \{0\}$. The quantity q_t is nonincreasing. Conditional on the first $t-1$ observations, the rank increases by at least one with probability exactly q_{t-1} and otherwise does not change. Define $Z_t = 2^{p-r_t}\mathbb{1}[q_t > \alpha]$. Then

$$\mathbb{E}[Z_t \mid x_1, \dots, x_{t-1}] \le \mathbb{1}[q_{t-1} > \alpha] 2^{p-r_{t-1}}(1 - q_{t-1}/2) \le (1-\alpha/2)Z_{t-1}.$$

Iterating gives $\mathbb{E}Z_m \le 2^p(1-\alpha/2)^m$. Since $r_m \le p$, $Z_m \ge \mathbb{1}[q_m > \alpha]$, proving the probability bound. Interpolation makes $\widehat{\theta} - \theta^\star$ orthogonal to V_m, so every equation whose vector lies in V_m holds for $\widehat{\theta}$. This proves the consequence. The same coverage event works simultaneously for any number of target coefficient vectors sharing these covariates. □

C.3 PROOF OF THEOREM 2

Theorem (Theorem 2: Exact master theorem)**.** *Consider a gated linear program of the form in Equation 1 whose training trace reveals blocks $s_1, \dots, s_M$ (gates or hidden states), in computation order, together with the label $f^\star(x)$. Suppose every block s_q admits an exact-recovery learner on an admissible feature map U_q, and that an admissible feature map F satisfies $f^\star(x) = \langle \beta^\star, F(x, s_{1:M}(x))\rangle$ for all x, for some $\beta^\star \in \mathbb{R}^D$. Then, for every $\varepsilon, \delta \in (0,1)$, there is an improper learner that uses only the input at test time and, with probability at least $1-\delta$, satisfies*

$$\Pr_{x\sim\mathcal{D}}\big[\widehat{f}(x) \ne f^\star(x)\big] \le \varepsilon.$$

If the feature maps are computable in polynomial time, the sample complexity and running time are polynomial in M, D, $1/\varepsilon$, and $\log(1/\delta)$, together with the costs of the local gate learners.

Proof. Set $\alpha = \varepsilon/(2M)$ and $\rho = \delta/(2M)$. Fit every local predictor on true covariates. Each oracle has its stipulated marginal i.i.d. sample distribution; sharing an example across different gates does not change that marginal. Independence between the fitted gate predictors is unnecessary. A union bound gives an event $\mathcal{E}_g$, of probability at least $1-\delta/2$, on which every event

$$B_q = \{x : \widehat{g}_q(U_q(x, s_{<q}(x))) \ne s_q(x)\}$$

has probability at most α.

At test time recursively set $\widehat{s}_q(x) = \widehat{g}_q(U_q(x, \widehat{s}_{<q}(x)))$ and $\widehat{F}(x) = F(x, \widehat{s}_{1:M}(x))$. If $x \notin \bigcup_q B_q$, induction on q shows $\widehat{s}_q(x) = s_q(x)$ for every q: preceding gates agree, so the constructor receives precisely its true arguments. Therefore

$$\Pr_x[\widehat{F}(x) \ne F(x, s_{1:M}(x))] \le \sum_{q=1}^{M} \Pr_x[B_q] \le \varepsilon/2 \quad \text{on } \mathcal{E}_g.$$

This argument requires no independence of the mistake events and places no restriction on the predictor after its first mistake.

On the independent readout batch solve

$$\langle \widehat{\beta}, F(x_i, s_{1:M}(x_i)) \rangle = f^\star(x_i), \qquad i \in [m_{\text{out}}].$$

The true β is feasible. Apply Lemma 1 to singleton sets containing these D-dimensional true features, with $\alpha_y = \varepsilon/2$ and $\rho_y = \delta/2$. With probability at least $1 - \delta/2$,

$$\Pr_x[\langle \widehat{\beta}, F(x, s_{1:M}(x)) \rangle \neq f^\star(x)] \leq \varepsilon/2.$$

The final predictor is $\widehat{f}(x) = \langle \widehat{\beta}, \widehat{F}(x) \rangle$. Its error event is contained in the union of feature disagreement and readout disagreement on true features. Another union bound proves the claimed error and confidence.

For explicit accounting, let C_U be the cost of forming all true local covariates on one fully observed example, C_F the cost of forming the final true feature, and T_q the cost of its local oracle on the data actually supplied. Training costs at most

$$O(m_{\text{gate}} C_U + m_{\text{out}} C_F + m_{\text{out}} D^2 + D^3) + \sum_q T_q.$$

Gate learners may use only the first m_q examples in the shared batch. At prediction the cost is that of the sequential constructors and local predictors, final feature formation, and $O(D)$ for the dot product. These quantities, including local prediction cost, must be polynomial for an efficient end-to-end learner. No hidden dependence on teacher weight magnitudes enters this reduction. The case $M = 0$ consists only of the span-lemma readout and is handled separately without dividing by M. □

C.4 Local exact-recovery oracles

Corollary (Local exact-recovery oracles)**.** *Let $U \in \mathbb{R}^p$ be an observed covariate and let $\alpha, \rho \in (0, 1)$. From noiseless i.i.d. examples, the following targets admit exact recovery outside probability α, with confidence $1 - \rho$: (i) $\mathbb{1}[w^\top U > 0]$; (ii) $\psi(BU)$ for a known injective link with an evaluable inverse; (iii)* $\mathrm{ReLU}(w^\top U)$*; and (iv)* $\mathrm{softmax}(BU)$ *when the whole row is observed. Cases (i) and (iii) use $O((p \log(2/\alpha) + \log(2/\rho))/\alpha)$ examples; cases (ii) and (iv) use $O((p + \log(2/\rho))/\alpha)$. The algorithms solve polynomial-size linear programs or systems; runtime includes the output dimension and link evaluations under the stated arithmetic convention.*

We use the standard realizable consistency guarantee for binary classes: for a measurable class of VC dimension at most v, independent examples with deterministic labels realized by the class, and any sample-consistent hypothesis, a sample size $C(v \log(2/\alpha) + \log(2/\rho))/\alpha$ suffices for disagreement at most α with confidence $1 - \rho$ (Blumer et al., 1989). The guarantee is distribution-free and imposes no margin or moment condition. We use it only for homogeneous halfspaces on *true* covariates in $\mathbb{R}^p$, whose VC dimension is at most p.

Proof. For sign gates, solve $\langle w, U_i \rangle \geq 1$ when the observed label is one, and $\langle w, U_i \rangle \leq 0$ otherwise. A realizing vector can be scaled to meet the positive constraints because the finite sample has a strictly positive minimum positive score; if there are no positives, use zero. The VC guarantee gives case (i).

For an injective ψ, each noiseless observation gives the entire vector $BU_i = \psi^{-1}(s_i)$. Solve these equations for $\widehat{B}$. By Lemma 1, with $O((p + \log(2/\rho))/\alpha)$ samples, U is in their span except on mass α. On that span every row of $\widehat{B}$ agrees with its target row simultaneously, so $\psi(\widehat{B}U) = \psi(BU)$. This is prediction recovery even if B is unidentifiable off the support.

For numerical ReLU observations $v_i = \sigma(\langle w^\star, U_i \rangle)$, solve

$$\langle \widehat{w}, U_i \rangle = v_i \quad (v_i > 0), \qquad \langle \widehat{w}, U_i \rangle \leq 0 \quad (v_i = 0).$$

The true vector is feasible. The binary classifier $\mathbb{1}[\langle \widehat{w}, U \rangle > 0]$ is sample-consistent with $\mathbb{1}[v > 0]$. The VC guarantee bounds its error by $\alpha/2$ with confidence $1 - \rho/2$. Separately define $Z = U\mathbb{1}[v > 0]$. The equations imply $\langle \widehat{w}, Z_i \rangle = \langle w^\star, Z_i \rangle = v_i$. The span lemma, on these i.i.d. Z_i, bounds $\Pr[v > 0, \langle \widehat{w}, U \rangle \neq v] \leq \alpha/2$ with confidence $1 - \rho/2$. If the target is zero, prediction can fail

only when the sign classifier fails; if it is positive, failure is covered by the latter span event. The union bound proves case (iii). This uses the same batch for both events and does not condition on a possibly rare positive class.

Finally, for $s = \mathrm{softmax}(BU)$, all entries are strictly positive, and

$$\log(s_i/s_k) = (B_i - B_k)U, \qquad i < k.$$

Interpolate these $k-1$ linear targets. Set the reconstructed reference logit to zero and the others to the fitted log-ratios, then apply softmax. On the span-coverage event all log-ratios are correct, hence the entire probability vector is correct. The true reference logit need not be zero: subtracting it from every logit leaves softmax unchanged. Runtime includes reading all output coordinates and evaluating exact inverse links; the sample dimension p alone is not a claim of output-dimension-free runtime. □

C.5 Proof of Corollary 2

Corollary (Corollary 2: ReLU networks)**.** *A depth-L, width-at-most-n, bias-free ReLU network with a linear readout is distribution-free PAC learnable from noiseless ordered sign gates and labels, with $O\big(\frac{nL}{\varepsilon}\big(dn^{L-1}\log\frac{2nL}{\varepsilon} + \log\frac{2nL}{\delta}\big)\big)$ examples. The learner runs in time polynomial in the path dimension dn^L.*

Proof. Pad each layer to width n with identically zero units. Proposition 1 gives $h^{(\ell-1)} = B_{\ell-1}\Phi^{(\ell-1)}$ with dimension $dn^{\ell-1}$. Thus each preactivation is a linear form in $\Phi^{(\ell-1)}$, and its revealed sign is a halfspace label on that true covariate. There are $M = nL$ gates and a final feature dimension $D = dn^L$. Substitute the sign-oracle bound with $\alpha = \varepsilon/(2nL)$ and $\rho = \delta/(2nL)$ into Theorem 2. The largest local dimension is dn^{L-1}. The resulting shared gate-batch bound is the displayed corollary bound. It absorbs the $O((dn^L + \log(2/\delta))/\varepsilon)$ readout batch because $nL\,dn^{L-1} = Ldn^L$ and the logarithms are bounded below by a positive constant. The construction uses polynomial-size LPs and linear systems. For rational data the usual bit-complexity bound also includes the data encoding length. No norm bound is required for the statistical result. □

C.6 Proof of Corollary 3

Why the attention lift works. Recall the features

$$z_t^{(0)} = x_t, \qquad z_t^{(\ell)} = \left(z_t^{(\ell-1)}, \sum_i p_{\ell,t,i} z_i^{(\ell-1)}\right).$$

We first show that these features retain everything needed from the hidden states. Specifically, for every layer ℓ there is a fixed matrix B_ℓ such that

$$h_t^{(\ell)} = B_\ell z_t^{(\ell)} \qquad \text{for every token } t.$$

Indeed, $B_0 = I$, and if $h_t^{(\ell-1)} = B_{\ell-1} z_t^{(\ell-1)}$, then

$$\begin{aligned} h_t^{(\ell)} &= h_t^{(\ell-1)} + A_\ell \sum_i p_{\ell,t,i} h_i^{(\ell-1)} \\ &= B_{\ell-1} z_t^{(\ell-1)} + A_\ell B_{\ell-1} \sum_i p_{\ell,t,i} z_i^{(\ell-1)} \\ &= \left[B_{\ell-1} \quad A_\ell B_{\ell-1}\right] z_t^{(\ell)}. \end{aligned}$$

Thus one may take

$$B_\ell = \left[B_{\ell-1} \quad A_\ell B_{\ell-1}\right].$$

In particular, the final output is linear in $z_N^{(L)}$.

Learning the attention scores. At layer ℓ, write

$$r = d2^{\ell-1}$$

and define

$$C_\ell^\star = B_{\ell-1}^\top W_\ell B_{\ell-1} \in \mathbb{R}^{r\times r}.$$

Then

$$b_{\ell,t,i} = (z_t^{(\ell-1)})^\top C_\ell^\star z_i^{(\ell-1)}.$$

Hence the score is linear in the r^2-dimensional quadratic feature

$$z_t^{(\ell-1)} \otimes z_i^{(\ell-1)}.$$

The learner need not recover $B_{\ell-1}$, W_ℓ, or the teacher's hidden states separately; their effect is absorbed into the single unknown matrix $C_\ell^\star$.

Hardmax rows. Suppose the revealed hardmax winner for query token t is $i^\star$. Then

$$b_{\ell,t,i^\star} > b_{\ell,t,i} \qquad \text{for every } i \neq i^\star,$$

or equivalently,

$$(z_t^{(\ell-1)})^\top C_\ell^\star \big(z_{i^\star}^{(\ell-1)} - z_i^{(\ell-1)}\big) > 0.$$

Thus learning the hardmax route is a multiclass linear classification problem in the unknown entries of $C_\ell^\star$.

Softmax rows. Suppose instead that the complete softmax row $p_{\ell,t,1}, \dots, p_{\ell,t,N}$ is revealed. Since every entry is strictly positive, for any reference coordinate, say N,

$$\log \frac{p_{\ell,t,i}}{p_{\ell,t,N}} = b_{\ell,t,i} - b_{\ell,t,N}.$$

Therefore

$$\log \frac{p_{\ell,t,i}}{p_{\ell,t,N}} = (z_t^{(\ell-1)})^\top C_\ell^\star \left(z_i^{(\ell-1)} - z_N^{(\ell-1)}\right), \qquad i < N.$$

Thus each revealed softmax row supplies $N-1$ linear equations in the entries of $C_\ell^\star$. Once these log-ratios are recovered, the entire softmax row is recovered exactly, since softmax is invariant to adding the same constant to all logits.

Corollary (Corollary 3: Residual attention)**.** *Consider the L-layer residual attention model in Equation 2 with N tokens and a linear readout. From noiseless complete hardmax routes or noiseless complete softmax attention rows, together with labels, the model is distribution-free PAC learnable by an improper learner that uses only the input at test time. For hardmax attention, it suffices to use $O\big(\frac{LN}{\varepsilon}\big(d^2\,4^{L-1}\log\frac{2NL}{\varepsilon} + \log\frac{2LN}{\delta}\big)\big)$ examples, and for softmax attention $O\big(\frac{LN}{\varepsilon}\big(d^2\,4^{L-1} + \log\frac{2LN}{\delta}\big)\big)$ examples, plus $O\big((d\,2^L + \log(1/\delta))/\varepsilon\big)$ examples for the readout. Under exact arithmetic (including logarithms and exponentials for softmax), the algorithms run in time polynomial in d, N, 2^L, $1/\varepsilon$, and $\log(1/\delta)$.*

Proof. Induction in equation 3 gives $h_t^{(\ell)} = B_\ell z_t^{(\ell)}$ with

$$B_0 = I_d, \qquad B_\ell = [B_{\ell-1} \;\; A_\ell B_{\ell-1}],$$

simultaneously for all tokens. This identity uses that the same A_ℓ acts on every source value at layer ℓ; the $p_{\ell,t,i}$ can otherwise be arbitrary. The output is therefore linear on $z_N^{(L)} \in \mathbb{R}^{d2^L}$.

Fix a layer and query token and put $r = d2^{\ell-1}$ and $C^\star = B_{\ell-1}^\top W_\ell B_{\ell-1} \in \mathbb{R}^{r\times r}$. The scores equal $z_t^\top C^\star z_i$. For hardmax, if the revealed winner is $i^\star$, require

$$z_t^\top C(z_{i^\star} - z_i) \geq 1 \quad (i \neq i^\star)$$

on every training example. These inequalities are feasible almost surely: uniqueness makes each of the finitely many true score gaps positive, allowing a common positive rescaling. A fixed tie rule defines the learned predictor on all inputs.

Here is the statistical bound needed for these multiclass fits. If predictions are argmaxes of N linear scores in a common p-dimensional parameter, then on m fixed inputs their label patterns are determined by at most $m\binom{N}{2}$ hyperplanes in parameter space. Counting all faces, including ties, bounds the number of patterns by

$$\Pi(m) \le \left(C_0 \max\{1, mN^2/p\}\right)^p .$$

To see the dependence, the total number of faces of an arrangement of H hyperplanes in $\mathbb{R}^p$ is at most $\sum_{j=0}^{p} \binom{H}{j} 2^j$; this follows by adding a hyperplane and counting the faces it splits in dimensions up to p. For $H \ge p$ the binomial sum is at most $(C_0 H/p)^p$; for $H < p$ it is at most $3^H \le 3^p$.

For completeness, the realizable double-sample argument bounds the probability that any sample-consistent classifier has error greater than α by $2\Pi(2m)\exp(-m\alpha/4)$ for $m\alpha$ larger than a universal constant. Indeed, on an independent ghost sample a fixed classifier of error exceeding α has at least $m\alpha/2$ mistakes with probability at least one half, by the binomial Chernoff bound. Conditional on the pooled $2m$ inputs, a random split can put all of any $k \ge m\alpha/2$ error positions in the ghost half with probability at most 2^{-k}. A union bound over at most $\Pi(2m)$ patterns gives the displayed, slightly looser exponential bound. The selected bad classifier can be fixed from the first sample before drawing the ghost sample, which justifies the first factor of two.

Consequently $m = C_1[p\log(2N/\alpha) + \log(2/\rho)]/\alpha$ is sufficient. To check the substitution, write $u = \log(2N/\alpha)$ and $v = p^{-1}\log(2/\rho)$. The logarithm of the growth bound is at most $p\log(C_2 N^2(u+v)/\alpha)$, which is $O(p(u+v))$ since $u \ge \log 2$ and $\log(u+v) \le u+v$. Increasing C_1 makes the negative exponential dominate this term and $\log(2/\rho)$. Apply this with $p = r^2$.

For softmax, noiseless complete attention rows give

$$\log \frac{p_{\ell,t,i}}{p_{\ell,t,N}} = z_t^\top C^\star (z_i - z_N), \qquad i < N.$$

Fit C by these linear equations. Apply the block version of Lemma 1 with

$$\mathcal{V}(X) = \{\mathrm{vec}(z_t(z_i - z_N)^\top) : i < N\} \subseteq \mathbb{R}^{r^2}.$$

All log-ratios are simultaneously correct outside mass α using $O((r^2 + \log(2/\rho))/\alpha)$ examples, independently of how many equations one example supplies. Correct log-ratios reconstruct the complete attention row exactly. No full-rank assumption is needed.

There are $M = LN$ route or attention-row blocks. Train every local fit on true z features computed from revealed earlier gates, then substitute their bounds at $\alpha = \varepsilon/(2LN)$ and $\rho = \delta/(2LN)$ into Theorem 2. Its readout feature is $z_N^{(L)}$. This proves both sample bounds and the replacement of the original $1/\delta$ readout dependence by $\log(1/\delta)$.

For example, forming all soft attention lifts on one fully observed trace uses at most $O(N^2 d 2^L)$ arithmetic operations. Each gate fit has at most $d^2 4^{L-1}$ unknown coefficients and $m(N-1)$ equations or inequalities; there are LN fits. Evaluating every learned score is polynomial in the same quantities. Thus all data construction and prediction costs are accounted for. Softmax uses exact logarithms and exponentials; hardmax uses polynomial-size LPs (polynomial bit time for rational data, including their encoding length). If $N = 1$, the gates are known constants and only a linear readout is needed. □

C.7 Architectural instances and exclusions

The table lists common architectures that fit Equation 1 and, for each, the local problem Theorem 2 must solve: what gate is revealed, the dimension of its local covariate, and the routine that recovers it exactly. Here $p = \dim \Phi^{(\ell-1)}$ for a generic program and $r = d2^{\ell-1}$ for the residual attention model.

Instance	Revealed gate and local form	Local dimension and guarantee
ReLU	$\mathbb{1}[w^\top U > 0]$	p; sign LP
Sigmoid GLU	$\psi(w^\top U)$, $\psi(t) = (1+e^{-t})^{-1}$	p; inverse-link interpolation
ReGLU	$\sigma(w^\top U)$, multiplying separate linear content	p; numerical ReLU LP
SwiGLU	$\mathrm{swish}(w^\top U)$	p; representation only without another oracle
Softmax mixture of linear experts	Complete router $\mathrm{softmax}(BU)$	p covariates; $(k-1)p$ logit-difference coefficients
Residual hardmax attention	Winner of $z_t^\top C z_i$	r^2 shared score coefficients per learned row
Residual softmax attention	Complete row $\mathrm{softmax}_i(z_t^\top C z_i)$	r^2 shared score coefficients per learned row

Justification of the table. For coordinatewise units $s_j = \psi(w_j^\top h)$, the path identity gives $w_j^\top h = w_j^\top B\Phi$, so the local covariate is $U = \Phi$. Multiplying s_j by a separate affine or linear content coordinate is exactly equation 1, after the constant-coordinate construction if needed. For mixtures of *linear* experts, expert outputs are $A_j h$ and the complete router provides their scalar multipliers; a nonlinear expert is not covered without being separately decomposed and supervised. Sigmoid inverts as $\log(s/(1-s))$. The local-oracle corollary and attention proof supply the remaining positive entries.

For standard swish $v(t) = t/(1+e^{-t})$, put $q(t) = (1+e^{-t})^{-1}$. Its derivative is $q(t)[1 + t(1 - q(t))]$. It is negative at $t = -2$ because $q(-2) < 1/2$, and positive at $t = 0$. A continuous injective real function on an interval must be strictly monotone, so swish is not globally injective. Its architectural inclusion therefore gives no inverse-link oracle. Restricting to a known injective branch could change that conclusion but is an additional assumption. □

D Approximate recovery: proof and computational requirements

D.1 Precise oracle contract

Fix all earlier fitted predictors before drawing the batch for layer ℓ. A robust oracle takes pairs $(\widehat{\Phi}^{(\ell-1)}(x_i), \widetilde{s}^{(\ell)}(x_i))$ with independent examples. For every preceding reconstruction with L_2 error at most r, it uses at most $m_\ell(r,\eta,\rho)$ examples and time $T_\ell(r,\eta,\rho)$ and, with conditional probability at least $1-\rho$, returns a predictor of the *clean* target satisfying

$$\|\widehat{s}^{(\ell)} - s^{(\ell)}\|_{L_2} \le K_\ell r + \eta.$$

The constants, dimensions, and permitted label-noise model are fixed as part of the contract. Taking a supremum of a bound over smaller r makes the sample and time functions nondecreasing if necessary. The range bound G_ℓ must hold for predicted and true gate vectors. Projection onto a closed convex known range containing the target is nonexpansive relative to the target and enforces this bound without increasing gate error.

One sufficient way to supply the contract is an agnostic regression algorithm satisfying

$$\|\widehat{g}(Z) - S\|_{L_2} \le \inf_{g \in \mathcal{G}_\ell} \|g(Z) - S\|_{L_2} + \eta$$

on the reconstructed covariate Z, together with a teacher $g_\ell^\star \in \mathcal{G}_\ell$ obeying

$$\|g_\ell^\star(\widehat{\Phi}^{(\ell-1)}) - g_\ell^\star(\Phi^{(\ell-1)})\|_{L_2} \le K_\ell \|\widehat{\Phi}^{(\ell-1)} - \Phi^{(\ell-1)}\|_{L_2}.$$

Substituting this comparator proves the robust guarantee by the triangle inequality. For $g_\ell^\star(z) = \psi(B_\ell^\star z)$, a κ_ℓ-Lipschitz link and $\|B_\ell^\star\|_{\mathrm{op}} \le \Lambda_\ell$ suffice with $K_\ell = \kappa_\ell \Lambda_\ell$. This verifies the *analytic* part of the contract, not the existence of an efficient agnostic optimizer for every link. Squared-loss optimization with a sigmoid or a ReLU parameterization is not asserted to be convex. Hard sign gates have no such global Lipschitz modulus without extra assumptions.

D.2 Proof of Theorem 3

Throughout, let

$$R_\ell = R_0 \prod_{r=1}^{\ell} G_r, \qquad c_\ell = G_\ell + R_{\ell-1} K_\ell,$$

so that R_ℓ bounds $\|\Phi^{(\ell)}\|_2$ and c_ℓ is the factor by which layer ℓ can amplify an incoming feature error. The *error-amplification factor* is

$$\mathcal{A}_L = \Theta \sum_{t=1}^{L} R_{t-1} \prod_{r=t+1}^{L} c_r, \tag{5}$$

with empty products equal to one.

Theorem (Theorem 3: Approximate master theorem)**.** *Consider a gated linear program of the form in Equation 1 whose training examples include clean labels and possibly corrupted gate observations, and let $X \sim \mathcal{D}$ denote a fresh input. Suppose that, for some known positive constants R_0, G_ℓ, and Θ, we have $\|X\|_2 \le R_0$ and $\|s^{(\ell)}(X)\|_2, \|\widehat{s}^{(\ell)}(X)\|_2 \le G_\ell$ for every $\ell \in [L]$, almost surely, and $\|\beta^\star\|_2 \le \Theta$. Suppose that, for any requested accuracy $\eta > 0$ and confidence $\rho > 0$, each layer admits a local learner that, trained on fresh reconstructed features after the earlier layers have been learned, satisfies Equation 4 with probability at least $1 - \rho$; and that a readout learner, trained on a fresh batch after all gates have been learned, returns $\widehat{\beta}$ whose L_2 error on $\widehat{\Phi}^{(L)}$ is within η of the best linear readout of norm at most Θ, with probability at least $1 - \rho$.*

Then, for every $\varepsilon, \delta \in (0,1)$, running all learners with $\eta = \varepsilon/(2\max\{1, \mathcal{A}_L\})$ and $\rho = \delta/(2L)$ yields an improper predictor $\widehat{f}(x) = \langle \widehat{\beta}, \widehat{\Phi}^{(L)}(x) \rangle$ that uses only x at test time and satisfies $\|\widehat{f} - f^\star\|_{L_2} \le \varepsilon$ with probability at least $1 - \delta$. If the path features are computable in polynomial time, the sample complexity and running time are polynomial in L, the path-feature dimensions, $1 + \mathcal{A}_L$, $1/\varepsilon$, and $\log(1/\delta)$, together with the costs of the local and readout learners.

Proof. Define deterministic upper bounds recursively by

$$q_0 = 0, \qquad e_\ell = K_\ell q_{\ell-1} + \eta_\ell, \qquad q_\ell = G_\ell q_{\ell-1} + R_{\ell-1} e_\ell = c_\ell q_{\ell-1} + R_{\ell-1}\eta_\ell.$$

Conditional on any successful preceding history, $\|\widehat{\Phi}^{(\ell-1)} - \Phi^{(\ell-1)}\|_{L_2} \le q_{\ell-1}$. The fresh-batch oracle therefore bounds the clean gate error by e_ℓ except with conditional probability ρ_ℓ. The tensor identity

$$\widehat{\Phi}^{(\ell)} - \Phi^{(\ell)} = (\widehat{s}^{(\ell)} - s^{(\ell)}) \otimes \Phi^{(\ell-1)} + \widehat{s}^{(\ell)} \otimes (\widehat{\Phi}^{(\ell-1)} - \Phi^{(\ell-1)})$$

and the Euclidean identity $\|u \otimes v\|_2 = \|u\|_2 \|v\|_2$ yield

$$\|\widehat{\Phi}^{(\ell)} - \Phi^{(\ell)}\|_{L_2} \le R_{\ell-1}\|\widehat{s}^{(\ell)} - s^{(\ell)}\|_{L_2} + G_\ell \|\widehat{\Phi}^{(\ell-1)} - \Phi^{(\ell-1)}\|_{L_2} \le q_\ell.$$

The first inequality uses pointwise bounds and Minkowski's inequality, not independence between factors. Pointwise $\|\Phi^{(\ell-1)}\|_2 \le R_{\ell-1}$ follows by induction from the gate and input bounds. Conditional failure bounds, summed over the first unsuccessful layer, show that all these conclusions hold with probability at least $1 - \sum_\ell \rho_\ell$.

Condition on such a history. The true coefficient β, with norm at most Θ, is a feasible comparator on the reconstructed final features, with risk

$$\|\langle \beta, \widehat{\Phi}^{(L)} \rangle - f^\star\|_{L_2} = \|\langle \beta, \widehat{\Phi}^{(L)} - \Phi^{(L)} \rangle\|_{L_2} \le \Theta q_L.$$

The independent readout oracle adds η_y and fails with conditional probability at most ρ_y. Finally, induction on the affine recurrence gives

$$q_L = \sum_{t=1}^{L} R_{t-1}\eta_t \prod_{r=t+1}^{L} c_r.$$

Thus

$$\|\widehat{f} - f^\star\|_{L_2} \le \Theta \sum_{t=1}^{L} R_{t-1}\eta_t \prod_{r=t+1}^{L} c_r + \eta_y \tag{6}$$

with confidence at least $1 - \sum_\ell \rho_\ell - \rho_y$. For the exact parameters of Theorem 3, take all $\eta_t = \eta_y = \varepsilon/(2\max\{1, \mathcal{A}_L\})$ and all $\rho_\ell = \rho_y = \delta/(2L)$. Then $(\mathcal{A}_L + 1)\eta \le \varepsilon$ and $(L+1)\delta/(2L) \le \delta$ for $L \ge 1$. This also covers $\mathcal{A}_L = 0$. Here ε is root mean-squared accuracy; squared risk at most ε instead uses this theorem with target $\sqrt{\varepsilon}$. □

D.3 SAMPLE AND TIME ACCOUNTING

The theorem uses the common tolerance and confidence above. An also-valid alternative is $\rho_\ell = \delta/(2L)$, $\rho_y = \delta/2$, $\eta_y = \varepsilon/2$, and $\eta_\ell = \varepsilon/(2\mathcal{A}_L)$ when $\mathcal{A}_L > 0$. With fresh batches,

$$m_{\text{total}} = \sum_{\ell=1}^{L} m_\ell(q_{\ell-1}, \eta_\ell, \delta/(2L)) + m_y(\eta_y, \delta/2). \tag{7}$$

The dimensions of the local inputs and outputs are $D_{\ell-1}$ and k_ℓ. If $C_{\ell-1}^{\text{roll}}$ is the cost of computing a reconstructed prefix on one example, training costs at most

$$\sum_{\ell=1}^{L} \left[T_\ell + m_\ell C_{\ell-1}^{\text{roll}} + O(m_\ell D_\ell)\right] + m_y C_L^{\text{roll}} + T_y.$$

This includes reevaluating earlier predictors on fresh batches, which cannot generally be charged only to the tensor-product dimension. Prediction costs $C_L^{\text{roll}} + O(D_L)$. A polynomial total bound requires polynomial local training *and prediction* costs, polynomially controlled bounds entering those routines, and polynomial $\mathcal{A}_L$, in addition to a manageable explicit representation. A fixed finite amplification constant can be exponentially large in L; being finite is not the needed polynomial bound. General nonlinear local error profiles need their own recurrence and may introduce worse accuracy dependence.

D.4 A CONCRETE READOUT SUBROUTINE

Lemma 2 (Bounded constrained least squares)**.** *Suppose $Z \in \mathbb{R}^D$, $\|Z\|_2 \le R$, and $|Y| \le B$ almost surely. Let $\Theta \ge 0$, $Q = B + \Theta R$, and $\eta > 0$. An empirical squared-loss minimizer over $\|\theta\|_2 \le \Theta$, computed to excess empirical objective $\eta^2/4$, satisfies*

$$\|\langle\widehat{\theta}, Z\rangle - Y\|_{L_2} \le \inf_{\|\theta\|_2 \le \Theta} \|\langle\theta, Z\rangle - Y\|_{L_2} + \eta$$

with probability at least $1 - \rho$ using

$$m = O\left(\frac{Q^4}{\eta^4}\left[D \log\left(1 + \frac{C\Theta RQ}{\eta^2}\right) + \log\frac{2}{\rho}\right]\right)$$

examples when $0 < \eta < Q$. Degenerate cases $Q = 0$ or $\eta \ge Q$ need no estimation.

Proof. The loss $\ell_\theta = (\langle\theta, Z\rangle - Y)^2$ belongs to $[0, Q^2]$ and $|\ell_\theta - \ell_{\theta'}| \le 2QR\|\theta - \theta'\|_2$. If $\Theta R = 0$, the class has a single prediction and the claim is immediate. Otherwise cover the Euclidean parameter ball by a net of radius $a = \eta^2/(32QR)$ and size at most $(1 + 2\Theta/a)^D$. (This standard volume bound follows by taking a maximal a-separated subset: disjoint radius-$a/2$ balls around its points lie in the radius-$\Theta + a/2$ ball.) Hoeffding's inequality and a union bound show that empirical and population loss differ by at most $\eta^2/8$ on the net with the displayed sample size. Replacing an arbitrary parameter by its net point changes each of the two losses by at most $2QRa = \eta^2/16$. Thus the uniform deviation over the ball is at most $\eta^2/4$.

The population risk of an $\eta^2/4$-approximate empirical minimizer is at most the best population risk plus $2(\eta^2/4) + \eta^2/4 \le \eta^2$. Taking square roots and using $\sqrt{a^2 + \eta^2} \le a + \eta$ proves the claim. The optimization problem is a convex quadratic over a Euclidean ball. It can be solved to the stated objective tolerance by convex optimization with its numerical encoding and tolerance costs included; no exact finite-operation claim for an irrational optimizer is needed. □

For Theorem 3, take $Z = \widehat{\Phi}^{(L)}$, $R = R_L$, and noiseless $B = \Theta R_L$. The feature function is fixed before the readout batch, so this lemma applies conditionally. The rate is conservative but explicit and polynomial; optimizing it is unnecessary for the composition claim.

D.5 Additive noise and the stability example

Proposition 2 (Conditional unbiasedness). *Let $S = S(x)$ be square integrable and let $\widetilde{S} = S + \xi$, where $\mathbb{E}[\xi \mid x] = 0$ and $\mathbb{E}\|\xi\|_2^2 < \infty$. Conditional on a previously learned feature function $Z = Z(x)$ independent of a fresh example, every square-integrable predictor $g(Z)$ satisfies*

$$\mathbb{E}\|g(Z) - \widetilde{S}\|_2^2 = \mathbb{E}\|g(Z) - S\|_2^2 + \mathbb{E}\|\xi\|_2^2.$$

Proof. Expand the square. Its cross term has expectation $\mathbb{E}\langle g(Z(x)) - S(x), \mathbb{E}[\xi \mid x]\rangle = 0$. Integrability follows from Cauchy–Schwarz. Independence of the fresh batch makes the argument valid conditional on all previous fitted functions. Unconditional $\mathbb{E}\xi = 0$ would not suffice: with a uniform sign X, $S = 0$, $\xi = X$, and $g(Z) = X$, it fails. Thus unbiased noise preserves population risk comparisons but neither supplies an efficient local algorithm nor removes the approximation error caused by $Z \neq \Phi$. Bounded noise permits the readout concentration proof with a larger B; finite variance alone does not justify that Hoeffding calculation. □

For the stability example, define the true recurrence $v_0 = 0$, $v_\ell = 2v_{\ell-1}$ and the perturbed recurrence $\widehat{v}_0 = 0$, $\widehat{v}_\ell = 2\widehat{v}_{\ell-1} + \eta$. The difference d_ℓ satisfies $d_0 = 0$, $d_\ell = 2d_{\ell-1} + \eta$, whence $d_\ell = (2^\ell - 1)\eta$ by induction. This can be written as a gated linear program on state $(v, 1)$: use one gate $s = 2v$, one constant gate, and linear content maps that respectively output $(s, 0)$ from the constant coordinate and preserve $(0, 1)$. Replacing the local gate rule by $2v + \eta$ gives the perturbed recurrence. For any fixed finite L, finite range bounds exist, but they do not provide a uniform polynomial amplification guarantee. This example concerns local approximation to a transition rule; it does not assert that every gate's error against the clean teacher on the rolled-out covariate remains η, nor that every learning algorithm suffers this worst case.

D.6 Proof of Corollary 4

Corollary (Corollary 4: Noisy linear gates). *Consider a gated linear program satisfying the bounds of Theorem 3, and suppose each gate is linear in the preceding path feature, $s^{(\ell)}(X) = B_\ell^\star \Phi^{(\ell-1)}(X)$ with $\|B_\ell^\star\|_{\mathrm{F}} \le \Lambda_\ell$. The observed gates are $\widetilde{s}^{(\ell)} = s^{(\ell)} + \xi_\ell$, where $\mathbb{E}[\xi_\ell \mid X] = 0$ and $\|\xi_\ell\|_2 \le \sigma_\ell$ almost surely, with known bounds $\Lambda_\ell, \sigma_\ell$.*

Then, for every $\varepsilon, \delta \in (0, 1)$, there is an improper learner that uses only the input at test time and, with probability at least $1 - \delta$, returns a predictor satisfying $\|\widehat{f} - f^\star\|_{L_2} \le \varepsilon$. Its sample complexity and arithmetic running time are polynomial in L, the path dimensions, R_0, G_ℓ, Λ_ℓ, σ_ℓ, Θ, $R_{\max} = \max_{0 \le r \le L} R_0 \prod_{t=1}^{r} G_t$, $1 + \mathcal{A}_L$, $1/\varepsilon$, and $\log(1/\delta)$, where $\mathcal{A}_L$ is given by Equation 5 with $K_\ell = \Lambda_\ell$.

Proof. Fix a layer ℓ and condition on every predictor fitted at earlier layers. Since the layer-ℓ batch is fresh, the reconstructed covariate $Z(X) = \widehat{\Phi}^{(\ell-1)}(X)$ is now a fixed function of X, independent of that batch. Projection of preceding predicted gates onto their radius-G_r Euclidean balls gives $\|Z(X)\|_2 \le R_{\ell-1}$ almost surely. Write $S(X) = s^{(\ell)}(X)$, let $D_{\ell-1}$ be the dimension of Z, and set

$$p_\ell = k_\ell D_{\ell-1}, \qquad H_\ell = \Lambda_\ell R_{\ell-1} + G_\ell + \sigma_\ell.$$

On the fresh batch, choose a matrix $\widehat{B}_\ell$ whose empirical squared loss is within $\alpha_\ell^2/4$ of the minimum over $\|B\|_{\mathrm{F}} \le \Lambda_\ell$:

$$\widehat{L}_\ell(B) = \frac{1}{m} \sum_{i=1}^{m} \|BZ(X_i) - \widetilde{s}^{(\ell)}(X_i)\|_2^2.$$

This is a convex quadratic optimization problem over a Euclidean ball in p_ℓ parameters. For any $\alpha_\ell > 0$ and $\rho_\ell \in (0, 1)$, a sufficient sample size is

$$m \ge C \frac{H_\ell^4}{\alpha_\ell^4} \left[p_\ell \log\left(1 + \frac{C\Lambda_\ell R_{\ell-1} H_\ell}{\alpha_\ell^2}\right) + \log \frac{2}{\rho_\ell} \right] \tag{8}$$

for a universal constant C; the cases of a singleton hypothesis class or identically zero loss are immediate.

For completeness, the uniform-convergence calculation behind equation 8 is as follows. Every feasible B satisfies $\|BZ\|_2 \le \Lambda_\ell R_{\ell-1}$, while $\|\widetilde{s}^{(\ell)}\|_2 \le G_\ell + \sigma_\ell$. Thus the squared loss lies in $[0, H_\ell^2]$ and is $2R_{\ell-1}H_\ell$-Lipschitz as a function of B in Frobenius norm. A net of radius $a = \alpha_\ell^2/(32R_{\ell-1}H_\ell)$ for the feasible Frobenius ball has at most $(1 + 2\Lambda_\ell/a)^{p_\ell}$ points. Hoeffding's inequality and a union bound make the empirical and population losses differ by at most $\alpha_\ell^2/8$ on the net under equation 8. Replacing any feasible matrix by a net point changes each loss by at most $2R_{\ell-1}H_\ell a = \alpha_\ell^2/16$. Hence, with conditional probability at least $1 - \rho_\ell$, the uniform deviation over the entire feasible ball is at most $\alpha_\ell^2/4$. On this event, the approximate empirical minimizer has population squared loss at most α_ℓ^2 above that of the feasible comparator $B_\ell^\star$.

Because $Z(X)$ and $S(X)$ are fixed functions of X after conditioning on the previous fits, and $\mathbb{E}[\xi_\ell \mid X] = 0$, the cross term in squared loss vanishes. For every feasible B,

$$\mathbb{E}\|BZ - \widetilde{s}^{(\ell)}\|_2^2 = \mathbb{E}\|BZ - S\|_2^2 + \mathbb{E}\|\xi_\ell\|_2^2.$$

The last term does not depend on B. Therefore the same population comparison holds for squared error against the clean gate vector:

$$\|\widehat{B}_\ell Z - S\|_{L_2}^2 \le \|B_\ell^\star Z - S\|_{L_2}^2 + \alpha_\ell^2.$$

Since $S = B_\ell^\star \Phi^{(\ell-1)}$ and $\|B_\ell^\star\|_{\mathrm{op}} \le \|B_\ell^\star\|_{\mathrm{F}} \le \Lambda_\ell$,

$$\|B_\ell^\star Z - S\|_{L_2} \le \Lambda_\ell \|\widehat{\Phi}^{(\ell-1)} - \Phi^{(\ell-1)}\|_{L_2}.$$

Taking square roots and using $\sqrt{a^2 + b^2} \le a + b$ for $a, b \ge 0$ gives

$$\|\widehat{B}_\ell Z - S\|_{L_2} \le \Lambda_\ell \|\widehat{\Phi}^{(\ell-1)} - \Phi^{(\ell-1)}\|_{L_2} + \alpha_\ell.$$

Define $\widehat{s}^{(\ell)}(X)$ by projecting $\widehat{B}_\ell Z(X)$ onto the radius-G_ℓ Euclidean ball. The clean target $S(X)$ lies in that ball, so projection cannot increase its pointwise distance from $S(X)$. Thus the displayed bound remains valid for $\widehat{s}^{(\ell)}$, while the range condition of Theorem 3 is enforced. This verifies equation 4 with $K_\ell = \Lambda_\ell$ and $\eta_\ell = \alpha_\ell$.

The argument applies conditionally at each layer because its batch is independent of all preceding fits. Assign failure probability $\rho_\ell = \delta/(2L)$ to each local fit and $\rho_y = \delta/2$ to the readout. Lemma 2 provides the latter on a fresh batch: indeed $\|\widehat{\Phi}^{(L)}\|_2 \le R_L$ and $|f^\star(X)| \le \Theta R_L$ almost surely. If $\mathcal{A}_L > 0$, choose $\alpha_\ell = \varepsilon/(2\mathcal{A}_L)$ and $\eta_y = \varepsilon/2$. Theorem 3 then gives $\|\widehat{f} - f^\star\|_{L_2} \le \varepsilon$ with probability at least $1 - \delta$. If $\mathcal{A}_L = 0$, its gate-error term vanishes and only the readout tolerance is needed. The local sample bound equation 8, the readout bound of Lemma 2, and the explicit costs of computing the path features give polynomial bounds in the listed parameters, including $R_{\max}$. In particular, $H_\ell \le \Lambda_\ell R_{\max} + G_\ell + \sigma_\ell$ and the readout bound uses $\Theta R_L \le \Theta R_{\max}$; products of the G_ℓ cannot be omitted from this accounting. Approximate convex optimization gives polynomial bit time when the numerical inputs and requested tolerances have polynomial encodings. □

E Learning from a fixed unknown gate order

Theorem 6 (Learning from unordered sign hints)**.** *There exists an improper learner for bias-free ReLU networks $f^\star : \mathbb{R}^d \to \mathbb{R}$ with L ReLU layers of width n and a linear readout, given labels and all $K = nL$ activation gates under a fixed unknown permutation. For every input distribution $\mathcal{D}$ and $\varepsilon, \delta \in (0,1)$, it uses $N = \mathrm{poly}\big(d2^K, \frac{1}{\varepsilon}, \log\frac{1}{\delta}\big)$ i.i.d. samples and outputs $\widehat{f} : \mathbb{R}^d \to \mathbb{R}$ such that $\Pr_S\Big[\Pr_{x\sim\mathcal{D}}[\widehat{f}(x) \ne f^\star(x)] \le \varepsilon\Big] \ge 1 - \delta$. On rational training data, its running time is polynomial in these parameters and the numerical input encoding length.*

Proof. Write $D_{\mathrm{u}} = d2^K$, and let $\pi : [K] \to [L] \times [n]$ be the fixed unknown bijection satisfying $s_q(x) = a_j^{(t)}(x)$ whenever $\pi(q) = (t, j)$. For $A \subseteq [K]$, define

$$\Psi_A(x) = \left(x_i \prod_{q\in S} s_q(x) \right)_{i\in[d],\, S\subseteq A} \in \mathbb{R}^{d2^{|A|}},$$

with the empty product equal to 1. For analysis, put

$$Q_t = \pi^{-1}(\{t\} \times [n]), \qquad Q_{\le t} = \bigcup_{r=1}^{t} Q_r, \qquad Q_{\le 0} = \varnothing.$$

Expanding the network along paths shows that every layer-t gate is a homogeneous halfspace in Ψ_A whenever $Q_{\le t-1} \subseteq A$. Indeed, each preactivation is a linear combination of input coordinates multiplied by gates from preceding layers, and every such product occurs in Ψ_A. The same expansion gives $\theta^\star \in \mathbb{R}^{D_{\mathrm{u}}}$ such that

$$f^\star(x) = \langle \theta^\star, \Psi_{[K]}(x) \rangle \qquad \text{for all } x. \tag{9}$$

Set $\eta = \varepsilon/(6K)$ and choose

$$m_{\mathrm{fit}} = \left\lceil \frac{C}{\eta} \left(D_{\mathrm{u}} \log \frac{1}{\eta} + \log \frac{6KL}{\delta} \right) \right\rceil, \qquad m_{\mathrm{val}} = \left\lceil \frac{1}{2\eta^2} \log \frac{6KL}{\delta} \right\rceil,$$

where C is a sufficiently large universal constant. All batches below are mutually independent.

Starting with $A_0 = \varnothing$, at round $t \in [L]$ the learner uses m_{fit} fresh samples to fit a consistent homogeneous halfspace over $\Psi_{A_{t-1}}$ for each inactive hint $q \notin A_{t-1}$, whenever one exists. Denote its normal by $\widehat{\alpha}_{q,t}$ and its classifier by

$$\widetilde{s}_{q,t}(x) = \mathbb{1}\left[\langle \widehat{\alpha}_{q,t}, \Psi_{A_{t-1}}(x) \rangle > 0 \right].$$

After all candidates are fixed, evaluate them on m_{val} fresh validation samples. Accept each candidate whose empirical error is at most 2η. Let B_t be the set of accepted indices and put $A_t = A_{t-1} \cup B_t$. If $A_L \neq [K]$, output the zero predictor.

Conditioning on the history preceding round t fixes A_{t-1}. The fitting sample remains independent of this history, and homogeneous halfspaces over $\Psi_{A_{t-1}}$ have VC dimension at most D_{u}. The realizable VC bound therefore gives conditional failure probability at most $\delta/(3KL)$ for each realizable target hint. A union bound over all rounds and hints gives an event $\mathcal{E}_{\mathrm{fit}}$ of probability at least $1 - \delta/3$ on which every such fitted classifier has population error at most η.

Conditioning additionally on the fitting sample fixes all candidates before validation. For each candidate, Hoeffding's inequality gives

$$\Pr[|\widehat{\mathrm{err}}_{\mathrm{val}} - \mathrm{err}| > \eta \mid \text{history and fitting sample}] \le 2e^{-2m_{\mathrm{val}}\eta^2} \le \frac{\delta}{3KL}.$$

Another union bound gives an event $\mathcal{E}_{\mathrm{val}}$ of probability at least $1 - \delta/3$ on which every validation error differs from its population error by at most η. These conditional bounds justify the union bounds despite the adaptive active sets.

On $\mathcal{E}_{\mathrm{fit}} \cap \mathcal{E}_{\mathrm{val}}$, induction gives $Q_{\le t} \subseteq A_t$ for every t. Indeed, if $Q_{\le t-1} \subseteq A_{t-1}$, every still-inactive layer-t gate is realizable over $\Psi_{A_{t-1}}$. Its fitted population error is therefore at most η, and its validation error is at most 2η, so it is accepted. The induction starts with $Q_{\le 0} = A_0 = \varnothing$. Consequently, $A_L = [K]$. Moreover, every accepted classifier has population error at most 3η, by its acceptance threshold and $\mathcal{E}_{\mathrm{val}}$.

On the branch $A_L = [K]$, let $\tau(q)$ be the acceptance round of hint q, and set $\widehat{\alpha}_q = \widehat{\alpha}_{q,\tau(q)}$. Define the ideal classifier

$$\widetilde{s}_q(x) = \mathbb{1}\left[\langle \widehat{\alpha}_q, \Psi_{A_{\tau(q)-1}}(x) \rangle > 0 \right].$$

At prediction time, evaluate the hints in acceptance-round order:

$$\widehat{s}_q(x) = \mathbb{1}\left[\langle \widehat{\alpha}_q, \widehat{\Psi}_{A_{\tau(q)-1}}(x) \rangle > 0 \right],$$

where

$$\widehat{\Psi}_A(x) = \left(x_i \prod_{q \in S} \widehat{s}_q(x) \right)_{i \in [d],\, S \subseteq A}.$$

This recursion is well-defined because each candidate uses only hints accepted in earlier rounds.

If $\widetilde{s}_q(x) = s_q(x)$ for every q, induction on acceptance rounds gives $\widehat{s}_q(x) = s_q(x)$ for every q. Consequently,

$$\{\widehat{\Psi}_{[K]}(x) \neq \Psi_{[K]}(x)\} \subseteq \bigcup_{q=1}^{K} \{\widetilde{s}_q(x) \neq s_q(x)\}.$$

Thus, on $\mathcal{E}_{\mathrm{fit}} \cap \mathcal{E}_{\mathrm{val}}$,

$$\Pr_{x\sim\mathcal{D}}[\widehat{\Psi}_{[K]}(x) \neq \Psi_{[K]}(x)] \leq 3K\eta = \frac{\varepsilon}{2}. \tag{10}$$

Reserve an independent readout batch $x_1, \ldots, x_m$, where

$$m = \left\lceil \frac{4}{\varepsilon}\left(D_{\mathrm{u}} \log 2 + \log \frac{3}{\delta}\right)\right\rceil.$$

It may be sampled in advance and is used only if $A_L = [K]$. Put $Z_i = \Psi_{[K]}(x_i)$, and choose any $\widehat{\theta}$ satisfying

$$\langle \widehat{\theta}, Z_i \rangle = f^\star(x_i) \qquad (i = 1, \ldots, m).$$

Such a vector exists by equation 9. For $i = 0, \ldots, m$, define

$$V_i = \mathrm{span}\{Z_1, \ldots, Z_i\}, \qquad V_0 = \{0\}, \qquad r_i = \dim V_i,$$

and

$$q_i = \Pr_{x\sim\mathcal{D}}[\Psi_{[K]}(x) \notin V_i],$$

where x is an independent fresh draw. Interpolation implies $\widehat{\theta} - \theta^\star \in V_m^\perp$, so

$$\Pr_{x\sim\mathcal{D}}[\langle \widehat{\theta}, \Psi_{[K]}(x)\rangle \neq f^\star(x)] \leq q_m. \tag{11}$$

To bound q_m, let $\mathcal{F}_i = \sigma(Z_1, \ldots, Z_i)$. The increasing spans give $q_i \leq q_{i-1}$, while independence gives

$$r_i - r_{i-1} = \mathbb{1}[Z_i \notin V_{i-1}], \qquad \Pr[Z_i \notin V_{i-1} \mid \mathcal{F}_{i-1}] = q_{i-1}.$$

Set

$$Y_i = 2^{D_{\mathrm{u}} - r_i}\, \mathbb{1}[q_i > \varepsilon/2].$$

Then

$$\begin{aligned}\mathbb{E}[Y_i \mid \mathcal{F}_{i-1}] &\leq \mathbb{1}[q_{i-1} > \varepsilon/2]\, 2^{D_{\mathrm{u}} - r_{i-1}} \left(1 - \frac{q_{i-1}}{2}\right) \\ &\leq \left(1 - \frac{\varepsilon}{4}\right) Y_{i-1}.\end{aligned}$$

The first inequality uses monotonicity of q_i and the fact that $2^{D_{\mathrm{u}} - r_i}$ halves precisely when the span gains a dimension. The second holds because either $Y_{i-1} = 0$ or $q_{i-1} > \varepsilon/2$.

Since $Y_0 \leq 2^{D_{\mathrm{u}}}$ and $Y_m \geq \mathbb{1}[q_m > \varepsilon/2]$, iteration yields

$$\Pr[q_m > \varepsilon/2] \leq \mathbb{E}[Y_m] \leq 2^{D_{\mathrm{u}}} \left(1 - \frac{\varepsilon}{4}\right)^m \leq \exp\!\left(D_{\mathrm{u}} \log 2 - \frac{m\varepsilon}{4}\right) \leq \frac{\delta}{3}.$$

Thus the event $\mathcal{E}_{\mathrm{read}} = \{q_m \leq \varepsilon/2\}$ has probability at least $1 - \delta/3$. On this event, every interpolating readout has error at most $\varepsilon/2$ in equation 11.

On the branch $A_L = [K]$, output

$$\widehat{f}(x) = \langle \widehat{\theta}, \widehat{\Psi}_{[K]}(x)\rangle.$$

On $\mathcal{E}_{\mathrm{fit}} \cap \mathcal{E}_{\mathrm{val}} \cap \mathcal{E}_{\mathrm{read}}$, this branch is taken, and

$$\begin{aligned}\Pr_{x\sim\mathcal{D}}[\widehat{f}(x) \neq f^\star(x)] \leq{}& \Pr_{x\sim\mathcal{D}}[\widehat{\Psi}_{[K]}(x) \neq \Psi_{[K]}(x)] \\ &+ \Pr_{x\sim\mathcal{D}}[\langle \widehat{\theta}, \Psi_{[K]}(x)\rangle \neq f^\star(x)] \\ \leq{}& \varepsilon.\end{aligned}$$

The three events hold simultaneously with probability at least $1 - \delta$.

The total sample count is bounded by

$$\begin{aligned} N &= L(m_{\mathrm{fit}} + m_{\mathrm{val}}) + m \\ &= O\left(\frac{LKD_{\mathrm{u}}}{\varepsilon}\log\frac{6K}{\varepsilon} + \frac{LK^2}{\varepsilon^2}\log\frac{6KL}{\delta}\right). \end{aligned}$$

The fitting confidence term and the readout terms are absorbed by the two displayed terms. Since $L \le K \le \log_2 D_{\mathrm{u}}$, this has the claimed polynomial form, with logarithmic confidence dependence.

Finally, every feature dimension is at most D_{u}, and at most KL halfspaces are fitted. Consistency is decided by linear feasibility, using $\langle w, z\rangle \ge 1$ on positive examples and $\langle w, z\rangle \le 0$ on negative examples. These constraints are equivalent to finite-sample consistency under the strict-positive gate convention: a consistent normal can be rescaled when positive examples exist, and $w = 0$ suffices otherwise. The readout is obtained by solving a consistent linear system. Feature construction, validation, and prediction also require polynomial work. For rational training data, feature coordinates are input coordinates multiplied by bits, so standard linear programming and linear algebra give the stated polynomial running time, including dependence on the numerical input encoding length. □

F HARDNESS PROOFS

F.1 PROOF OF THEOREM 4

Theorem (Theorem 4: Hardness with revealed discrete traces)**.** *Under the cryptographic assumptions of Kearns & Valiant (1994), neither of the following classes is distribution-free PAC learnable in polynomial time, even when the learner receives the complete discrete trace during training and prediction and produces a polynomial-time evaluable hypothesis:*

1. *Bias-free ReLU networks of input dimension d, width at most n, and L hidden layers with a linear readout;*
2. *Residual hardmax-attention networks of token dimension d, sequence length N, and depth L, with a linear readout.*

Here polynomial time is measured in the stated model parameters, $1/\varepsilon$, and $\log(1/\delta)$; the loss is output disagreement.

Proof. The two reductions below establish the two classes separately. In each, a polynomial-time input map produces the network input and its complete target-independent discrete trace, preserves labels, and has polynomial encoding length. Thus a learner for either class would weakly learn the Kearns–Valiant hard automata. The reductions also cover arbitrary polynomial-time evaluable improper hypotheses. □

F.2 PROOF OF THEOREM 4 PART 1.

Theorem (Hardness of learning from revealed gates)**.** *Under the Kearns–Valiant cryptographic assumptions, there is no distribution-free PAC learner for bias-free ReLU networks running in time* $\mathrm{poly}\left(d, n, L, \frac{1}{\varepsilon}, \log\frac{1}{\delta}\right)$*, even if the full ReLU activation pattern is revealed both during training and prediction. More precisely, no such learner can, for every distribution $\mathcal{D}$ and every depth-L, width-n network f_W that is $\{0,1\}$-valued on* $\mathrm{supp}(\mathcal{D})$*, output with probability at least $1-\delta$ a polynomial-time evaluatable hypothesis h satisfying* $\Pr_{x\sim\mathcal{D}}\left[h(x, G_W(x)) \ne f_W(x)\right] \le \varepsilon$.

Proof. Suppose, toward a contradiction, that such a learner $\mathcal{L}_{\mathrm{ReLU}}$ exists. We show that it yields a polynomial-time weak learner for polynomial-size acyclic deterministic finite automata, contradicting the Kearns–Valiant hardness theorem.

Let $T \ge 1$, and let A be an acyclic deterministic finite automaton on $\{0,1\}^T$ with at most $S = p(T)$ states, where p is the polynomial appearing in the Kearns–Valiant hardness theorem. By padding the state space if necessary, we may index the states by

$$Q = [S].$$

Let $q_0 \in Q$ be the initial state, let $F \subseteq Q$ be the accepting set, and let

$$\delta_b : Q \to Q, \qquad b \in \{0,1\},$$

denote the transition maps. For $x = (x_1, \ldots, x_T) \in \{0,1\}^T$, write

$$q_t = \delta_{x_t}(q_{t-1}), \qquad t = 1, \ldots, T,$$

so that

$$A(x) = \mathbf{1}\{q_T \in F\}.$$

For each $b \in \{0,1\}$, define $P_b \in \{0,1\}^{S \times S}$ by

$$P_b e_q = e_{\delta_b(q)}, \qquad q \in Q.$$

Encode state q by

$$z(q) := \mathbf{1} + e_q \in \mathbb{R}^S.$$

Every coordinate of $z(q)$ belongs to $\{1,2\}$ and is therefore strictly positive.

To obtain a bias-free construction, augment the input by a constant coordinate and write

$$\bar{x} = (c, x_1, \ldots, x_T), \qquad c = 1.$$

The first ReLU layer constructs

$$z_0 := c(\mathbf{1} + e_{q_0}), \qquad r_j := c + x_j, \quad j \in [T],$$

and carries c. On the support of the input distribution,

$$c = 1, \qquad r_j \in \{1,2\},$$

so all gates in this layer are active.

For $b \in \{0,1\}$ define the linear map

$$\tau_b(z, c) := P_b z + (\mathbf{1} - P_b \mathbf{1})c.$$

For every state q,

$$\tau_b(z(q), 1) = P_b(\mathbf{1} + e_q) + \mathbf{1} - P_b\mathbf{1} = \mathbf{1} + e_{\delta_b(q)} = z(\delta_b(q)).$$

Suppose that before processing the t-th bit the state block equals $z_{t-1} = z(q_{t-1})$. The next ReLU layer computes

$$a_t^0 = \sigma(\tau_0(z_{t-1}, c) - 3(r_t - c)\mathbf{1})$$

and

$$a_t^1 = \sigma(\tau_1(z_{t-1}, c) - 3(2c - r_t)\mathbf{1}),$$

while carrying $c, r_1, \ldots, r_T$ unchanged. All of these maps are linear in the preceding coordinates, so no biases are required.

If $x_t = 0$, then $r_t = c = 1$, and hence

$$a_t^0 = \tau_0(z_{t-1}, 1), \qquad a_t^1 = 0,$$

because every coordinate of $\tau_1(z_{t-1}, 1) - 3\mathbf{1}$ is either -2 or -1. If $x_t = 1$, then $r_t = 2$ and

$$a_t^0 = 0, \qquad a_t^1 = \tau_1(z_{t-1}, 1).$$

Thus, in either case, exactly one candidate block is strictly positive coordinatewise and the other is strictly negative before application of the ReLU.

A subsequent ReLU layer forms

$$z_t := \sigma(a_t^0 + a_t^1)$$

and again carries $c, r_1, \ldots, r_T$. Since the surviving candidate block is strictly positive,

$$z_t = z(\delta_{x_t}(q_{t-1})) = z(q_t).$$

Induction on t therefore gives

$$z_T = z(q_T).$$

The final linear output is

$$N_A(\bar{x}) := \sum_{q \in F} (z_T)_q - |F|c.$$

Since $c = 1$ and $z_T = \mathbf{1} + e_{q_T}$,

$$N_A(\bar{x}) = \begin{cases} 1, & q_T \in F, \\ 0, & q_T \notin F, \end{cases}$$

and consequently

$$N_A(\bar{x}) = A(x) \qquad \forall\, x \in \{0,1\}^T. \tag{12}$$

The network has depth $2T + 1$ and width at most

$$2S + T + 1,$$

up to harmless padding by neurons that are identically inactive. Since $S = p(T)$, its depth, width, input dimension, and total size are all polynomial in T.

Crucially, its activation pattern does not reveal the automaton. The initial state block, the constant coordinate, the bit-carrier coordinates, and every merged state block are always strictly positive. At time t, all coordinates of a_t^0 are active exactly when $x_t = 0$, whereas all coordinates of a_t^1 are active exactly when $x_t = 1$. Any padding neurons may be chosen to be always inactive. Hence there exists a deterministic, polynomial-time computable map

$$\Gamma_{S,T} : \{0,1\}^T \to \{0,1\}^M$$

such that

$$G_{N_A}(\bar{x}) = \Gamma_{S,T}(x) \qquad \forall\, x \in \{0,1\}^T, \tag{13}$$

and $\Gamma_{S,T}$ depends only on S, T and x, not on q_0, δ_0, δ_1, or F.

Let D_+ and D_- be arbitrary positive- and negative-example distributions for the target automaton and let $\mathcal{D} = (D_+ + D_-)/2$. A learner with access to both oracles can sample this mixture by a fair coin flip. Suppose we are given i.i.d. labeled examples

$$(x, A(x)), \qquad x \sim \mathcal{D}.$$

Map each example, in polynomial time, to

$$(\bar{x}, \Gamma_{S,T}(x), A(x))\,.$$

By equation 12 and equation 13, these are exactly revealed-gate examples for the ReLU network N_A. Importantly, this transformation does not require knowledge of the unknown automaton A.

Run $\mathcal{L}_{\mathrm{ReLU}}$ on these examples with, for instance,

$$\varepsilon = \frac{1}{8}, \qquad \delta = \frac{1}{3}.$$

By assumption, with probability at least $2/3$, it outputs a polynomial-time evaluatable hypothesis h satisfying

$$\Pr_{x \sim \mathcal{D}} [h(\bar{x}, G_{N_A}(\bar{x})) \neq N_A(\bar{x})] \leq \frac{1}{8}.$$

Define

$$\widehat{A}(x) := \mathbb{1}[h(\bar{x}, \Gamma_{S,T}(x)) > 1/2].$$

Then

$$\begin{aligned} \Pr_{x \sim \mathcal{D}} \left[\widehat{A}(x) \neq A(x)\right] &\leq \Pr_{x \sim \mathcal{D}} [h(\bar{x}, G_{N_A}(\bar{x})) \neq N_A(\bar{x})] \\ &\leq \frac{1}{8}. \end{aligned}$$

If e_+, e_- are its errors under D_+, D_-, then $(e_+ + e_-)/2 \leq 1/8$, so each is at most $1/4$. Thus $\widehat{A}$ achieves a constant advantage on both distributions, as required by the cited weak-learning model.

Moreover, because

$$d = T + 1, \qquad n = O(S + T), \qquad L = O(T),$$

and $S = p(T)$, both the invocation of $\mathcal{L}_{\mathrm{ReLU}}$ and the evaluation of $\Gamma_{S,T}$ and $\widehat{A}$ take time polynomial in T. The assumed ReLU learner would therefore give a polynomial-time distribution-free weak learner for the polynomial-size acyclic deterministic finite automata appearing in the Kearns–Valiant hardness theorem.

This contradicts the Kearns–Valiant cryptographic assumption. Hence no such polynomial-time learner from revealed ReLU activation patterns can exist. □

F.3 Proof of Theorem 4 part 2.

Theorem (Hardness from revealed hardmax routing)**.** *Under the Kearns–Valiant cryptographic assumptions, residual hardmax-attention networks are not distribution-free PAC learnable in time* $\mathrm{poly}\left(d, N, L, \frac{1}{\varepsilon}, \log \frac{1}{\delta}\right)$ *, even when the complete routing trace is supplied both during training and prediction. This holds even for networks whose hardmax decisions are unique and whose outputs are Boolean almost surely.*

Proof. Suppose, toward a contradiction, that such a learner $\mathcal{L}_{\mathrm{attn}}$ exists. We construct from it a polynomial-time weak learner for the class of polynomial-size acyclic deterministic finite automata appearing in the Kearns–Valiant hardness theorem.

Let p be the polynomial supplied by that theorem, fix an input length T, and set

$$S := p(T).$$

Let $\mathcal{A}$ be a target deterministic finite automaton of size at most S on $\{0,1\}^T$. By relabeling its states and padding with unreachable states, we may assume without loss of generality that its state set is $Q = [S]$ and its initial state is the known state

$$q_0 = 1.$$

Let $F \subseteq Q$ be the accepting states and $\delta_b : Q \to Q$, $b \in \{0,1\}$, the transition maps. For $x = (x_1, \ldots, x_T)$, write

$$q_t = \delta_{x_t}(q_{t-1}), \qquad \mathcal{A}(x) = \mathbf{1}\{q_T \in F\}.$$

Define

$$P_b e_q = e_{\delta_b(q)}, \qquad b \in \{0,1\}.$$

We construct a residual hardmax-attention network $\mathcal{N}_{\mathcal{A}}$. Let

$$M := 3T + 2$$

and introduce the tokens

$$z, \qquad s_0, s_1, \ldots, s_T, \qquad c_t^0, c_t^1 \quad (t \in [T]),$$

ordered so that s_T is token M. Fix a bijective address map a from these tokens to $[M]$ and set

$$d := S + 2M.$$

Write every hidden vector as

$$h = (h^{\mathrm{w}}, h^{\mathrm{id}}, h^{\mathrm{sel}}) \in \mathbb{R}^S \oplus \mathbb{R}^M \oplus \mathbb{R}^M.$$

For $x \in \{0,1\}^T$, define the input $X(x)$ by

$$h_z^{(0)} = (0, e_{a(z)}, 0), \qquad h_{s_0}^{(0)} = (e_1, e_{a(s_0)}, 0),$$

and, for $t \in [T]$,

$$h_{s_t}^{(0)} = (0, e_{a(s_t)}, e_{a(c_t^{x_t})}), \qquad h_{c_t^b}^{(0)} = (0, e_{a(c_t^b)}, 0), \quad b \in \{0,1\}.$$

Thus $X(x)$ depends only on x, S, and T, and not on the unknown transition maps or accepting set.

The network has depth $3T$. Choose each query-key matrix W_ℓ so that

$$(h_v^{(\ell-1)})^\top W_\ell h_i^{(\ell-1)} = (h_v^{\mathrm{id}})^\top \Pi_\ell h_i^{\mathrm{id}} + \mu_\ell (h_v^{\mathrm{sel}})^\top h_i^{\mathrm{id}},$$

and choose the value matrix to act only on the work block:

$$V_\ell h = (V_\ell^{\mathrm{w}} h^{\mathrm{w}}, 0, 0).$$

Hence the id and selector blocks remain unchanged throughout the network.

For $t \in [T]$, define

$$\mu_{3t-2} = 0, \qquad \Pi_{3t-2} = 2\left(e_{a(c_t^0)} e_{a(s_{t-1})}^\top + \sum_{v \neq c_t^0} e_{a(v)} e_{a(z)}^\top\right), \qquad V_{3t-2}^{\mathrm{w}} = P_0,$$

and

$$\mu_{3t-1} = 0, \qquad \Pi_{3t-1} = 2\left(e_{a(c_t^1)} e_{a(s_{t-1})}^\top + \sum_{v \neq c_t^1} e_{a(v)} e_{a(z)}^\top\right), \qquad V_{3t-1}^{\mathrm{w}} = P_1.$$

At layer $3t$, let

$$\mu_{3t} = 1, \qquad \Pi_{3t} = 2\sum_{v \neq s_t} e_{a(v)} e_{a(z)}^\top, \qquad V_{3t}^{\mathrm{w}} = I_S.$$

At layer $3t-2$, token c_t^0 uniquely attends to s_{t-1} and every other token uniquely attends to z. At layer $3t-1$, token c_t^1 uniquely attends to s_{t-1} and every other token uniquely attends to z. At layer $3t$, token s_t has zero id-based score and selector

$$e_{a(c_t^{x_t})},$$

so it uniquely attends to $c_t^{x_t}$. Every other token has score 2 for z, whereas its selector contribution to any other token is at most 1, so z remains its unique maximizer. Thus every hardmax decision is unique.

Moreover, these routing decisions depend only on the fixed addresses and on x_t, and not on P_0, P_1, or F. Consequently, there exists a polynomial-time computable function

$$\Gamma_{S,T} : \{0,1\}^T \to [M]^{3TM}$$

such that

$$\Pi_{\mathcal{N}_\mathcal{A}}(X(x)) = \Gamma_{S,T}(x) \qquad \forall x \in \{0,1\}^T, \tag{14}$$

and $\Gamma_{S,T}$ is independent of the target automaton.

We next verify the simulated computation. The zero token has work block identically zero. Before the layers associated with time t, the tokens c_t^0, c_t^1, s_t also have zero work blocks, since prior updates send them only values originating from z. If

$$h_{s_{t-1}}^{\mathrm{w}} = e_{q_{t-1}},$$

then the residual update at layer $3t-2$ gives

$$h_{c_t^0}^{\mathrm{w}} = P_0 e_{q_{t-1}} = e_{\delta_0(q_{t-1})},$$

and the update at layer $3t-1$ gives

$$h_{c_t^1}^{\mathrm{w}} = P_1 e_{q_{t-1}} = e_{\delta_1(q_{t-1})}.$$

At layer $3t$, token s_t attends to $c_t^{x_t}$ and therefore

$$h_{s_t}^{\mathrm{w}} = e_{\delta_{x_t}(q_{t-1})} = e_{q_t}.$$

Since

$$h_{s_0}^{\mathrm{w}} = e_1 = e_{q_0},$$

induction gives

$$h_{s_T}^{\mathrm{w}} = e_{q_T}.$$

Define the readout vector by

$$u = \left(\sum_{q \in F} e_q,\ 0,\ 0\right).$$

Since s_T is the final token,

$$\mathcal{N}_{\mathcal{A}}(X(x)) = \langle u, h_{s_T}^{(3T)}(X(x))\rangle = \mathbf{1}\{q_T \in F\} = \mathcal{A}(x).$$

The construction satisfies

$$N = M = 3T + 2, \qquad L = 3T, \qquad d = S + 2M = O(S + T),$$

and all its parameters have polynomial bit complexity.

It remains to derive the contradiction in the weak-learning model. Let D_+ and D_- be arbitrary distributions over the positive and negative examples of $\mathcal{A}$, respectively. Generate a labeled example by choosing $b \in \{0, 1\}$ uniformly and then sampling

$$x \sim \begin{cases} D_+, & b = 1, \\ D_-, & b = 0. \end{cases}$$

This defines a distribution D_0 on $\{0, 1\}^T$. Map every resulting example in polynomial time to

$$\big(X(x), \Gamma_{S,T}(x), \mathcal{A}(x)\big).$$

By the preceding construction, these are precisely revealed-routing examples for $\mathcal{N}_{\mathcal{A}}$ under the pushforward of D_0.

Run $\mathcal{L}_{\mathrm{attn}}$ with

$$\varepsilon = \frac{1}{8}, \qquad \delta = \frac{1}{3}.$$

Because

$$d = O(p(T) + T), \qquad N = O(T), \qquad L = O(T),$$

its running time is polynomial in T. With probability at least $2/3$, it returns a polynomial-time evaluable hypothesis, which can be thresholded at $1/2$ without increasing disagreement on Boolean targets. Denote the resulting Boolean hypothesis by h; it satisfies

$$\Pr_{x \sim D_0}\left[h(X(x), \Gamma_{S,T}(x)) \neq \mathcal{A}(x)\right] \leq \frac{1}{8}.$$

Define

$$\widehat{\mathcal{A}}(x) := h(X(x), \Gamma_{S,T}(x)).$$

If e_+ and e_- denote its errors under D_+ and D_-, respectively, then, by construction of D_0,

$$\frac{e_+ + e_-}{2} \leq \frac{1}{8}.$$

Since both errors are nonnegative,

$$e_+ \leq \frac{1}{4}, \qquad e_- \leq \frac{1}{4}.$$

Hence $\widehat{\mathcal{A}}$ has a constant advantage over random guessing on both the positive and negative distributions. This is a polynomial-time weak learner for the polynomial-size acyclic deterministic finite automata of the Kearns–Valiant hard class.

Under the Kearns–Valiant cryptographic assumptions, such a weak learner cannot exist. The assumed polynomial-time learner for residual hardmax-attention networks with revealed routing therefore cannot exist. □

G Polynomial-time learning under a path-margin assumption

G.1 Proof of Theorem 5

Theorem (Theorem 5: Efficient learning from sign traces under a path margin)**.** *Consider a bias-free ReLU network with widths $n_1, \ldots, n_L$ and real-valued linear readout $f^\star(x) = c^\top h^{(L)}(x)$; by Proposition 1, $f^\star(x) = \langle \beta, \Phi^{(L)}(x)\rangle$ with $\beta = B_L^\top c$. Let $M = \sum_{\ell=1}^{L} n_\ell$ be the number of hidden gates. Fix $\varepsilon, \delta \in (0, 1)$. Assume:*

1. Path margin *with known* $\gamma \in (0,1]$*: for every hidden gate* (ℓ, j)*, conditional on* $\Phi^{(\ell-1)}(X) \neq 0$*, either* $s_j^{(\ell)}(X)$ *is almost surely constant, or there is a unit vector* $u_{\ell,j}$ *such that, almost surely,* $\big(2s_j^{(\ell)}(X)-1\big)\big\langle u_{\ell,j},\, \Phi^{(\ell-1)}(X)/\|\Phi^{(\ell-1)}(X)\|\big\rangle \geq \gamma$.

2. Norm bound*:* $\|\beta\| \leq \Theta$ *and* $\Theta\,\|\Phi^{(L)}(X)\| \leq B$ *almost surely, with* Θ *and* B *known to the learner.*

Then a learner given ordered hidden activation signs and labels during training returns a predictor $\widehat{f}$ *that uses only* x *at test time and satisfies* $\|\widehat{f} - f^\star\|_{L_2} \leq \varepsilon$ *with probability at least* $1-\delta$*. Its sample complexity and arithmetic running time are polynomial in* d*,* M*,* B*,* $1/\gamma$*,* $1/\varepsilon$*, and* $\log(1/\delta)$.

Proof. We learn only the hidden signs by classification and learn the real-valued output by norm-constrained kernel regression. There is no output-margin assumption. If $B = 0$ or $\Theta = 0$, the target is zero almost surely. If $\varepsilon \geq B$, the zero predictor already suffices. Hence assume $0 < \varepsilon < B$ and $\Theta > 0$.

Hidden-gate kernel and normalization. For a supplied prefix $z = (x, s^{(1)}, \ldots, s^{(r)})$, write $\Phi_r(z) = s^{(r)} \otimes \cdots \otimes s^{(1)} \otimes x$ and

$$Q_r(z) = \|x\|^2 \prod_{a=1}^{r} \|s^{(a)}\|^2 = \|\Phi_r(z)\|^2.$$

When $Q_r(z) > 0$, choose the unique integer $b_r(z)$ with $2^{2(b_r(z)-1)} < Q_r(z) \leq 2^{2b_r(z)}$ and put $a_r(z) = 2^{b_r(z)}$. Then $\|\Phi_r(z)\| \leq a_r(z) < 2\|\Phi_r(z)\|$. Let $\psi_r = \Phi_r/a_r$ on nonzero features and $\psi_r = 0$ otherwise. Its kernel is

$$K_r^{\mathrm{dy}}(z, z') = \frac{\langle x, x'\rangle \prod_{a=1}^{r} \langle s^{(a)}, s'^{(a)}\rangle}{a_r(z)\, a_r(z')}, \tag{15}$$

with value zero when either feature vanishes. It is positive semidefinite, $\|\psi_r\| \leq 1$, and each non-constant gate has signed margin at least $\gamma/2$ in these features. All kernel evaluations use $O(d+M)$ arithmetic operations after normalization. In the exact-real arithmetic model, one may instead set $a_r = \sqrt{Q_r}$, counting square roots as elementary operations; this only improves the margin. The dyadic version gives the rational bit-time implementation discussed below.

Gate fitting and termination. Put $k = \lceil 4/\gamma^2 \rceil$. For each hidden gate (ℓ, j), use the true training prefixes $Z_i = (X_i, s^{(1:\ell-1)}(X_i))$ and labels $y_i = 2s_j^{(\ell)}(X_i) - 1$. At a zero query feature always return zero. This is correct on a true prefix: either $x = 0$ or a preceding all-zero sign vector makes the hidden state zero, and all later bias-free states and signs remain zero.

On nonzero training features, return the constant zero function if none occur, or the common label if all labels agree. Otherwise, initialize $w = 0$ and repeatedly sweep the sample in fixed order, updating $w \leftarrow w + y_i \psi_{\ell-1}(Z_i)$ whenever $y_i \langle w, \psi_{\ell-1}(Z_i)\rangle \leq 0$. Stop after a sweep without an update. If both labels occur, the constant alternative in the assumption is impossible, so a unit separator u with margin $\gamma/2$ exists. After t updates,

$$\langle w_t, u\rangle \geq t\gamma/2, \qquad \|w_t\|^2 \leq t.$$

The second inequality follows because the cross term at each update is nonpositive and $\|\psi\|^2 \leq 1$. Cauchy–Schwarz gives $t \leq 4/\gamma^2 \leq k$. There are at most $k+1$ sweeps, and the final predictor is sample-consistent. Store the ordered update indices, allowing repetitions; prediction sums their labeled kernel evaluations. The two constant candidates mean zero on zero features and the chosen constant on nonzero features.

Gate generalization. For one gate let $e(h)$ denote disagreement on a fresh true prefix. For any $\alpha \in (0,1)$ and $m \geq k$,

$$\Pr[e(h) > \alpha] \leq 3(m+1)^k e^{-\alpha(m-k)}. \tag{16}$$

Indeed, fix an ordered index sequence of length at most k. It determines a predictor using at most k distinct examples. Conditional on those examples, the remaining at least $m-k$ examples are

independent. If the candidate's error exceeds α, consistency on the remainder has probability at most $(1-\alpha)^{m-k}$. Union-bound over the fixed index sequences and the two constant candidates: there are at most $2+\sum_{t=0}^{k} m^t \le 3(m+1)^k$. The actual consistent predictor belongs to this family. No conditioning on the adaptively selected sequence is used.

Set $\alpha = \varepsilon^2/(8B^2M)$ and use confidence $\delta/(2M)$ for each hidden gate. For example it suffices to take

$$m_g = \left\lceil \frac{8k}{\alpha}\log\frac{4k}{\alpha} + \frac{4}{\alpha}\left(k + \log\frac{6M}{\delta}\right)\right\rceil. \tag{17}$$

To verify this choice, for $a \ge 1, b \ge 0$, the elementary inequality $x \ge 4a\log(2a) + 2b$ implies $x \ge a\log x + b$. For $x_0 = 4a\log(2a) + 2b$, if $b \le 2a\log(2a)$, use $\log x_0 \le \log(8a\log(2a)) \le 2\log a + \log 16 - 1$; otherwise use $\log x_0 < \log(4b) \le \log(4a) + b/a - 1$. These bounds give $x_0 - a\log x_0 \ge b$, and $x - a\log x$ is increasing for $x \ge a$. Apply this with $a = 2k/\alpha$ and $b = 2(k + \log(6M/\delta))/\alpha$; also $m_g \ge 2k$ and $\log(m_g+1) \le \log m_g + 1$. Substitution into equation 16 gives the claimed confidence. A union bound, permitting one shared gate batch, yields

$$\Pr_X[\widehat{s}^{(1:L)}(X) \ne s^{(1:L)}(X)] \le M\alpha = \varepsilon^2/(8B^2) \tag{18}$$

with training probability at least $1 - \delta/2$. This is the first-mistake induction: outside all true-prefix mistake events, every predicted prefix equals its true prefix.

Real-valued readout: a dimension-free bound. On an independent batch use the *true* final features and set $v(X) = \Theta\Phi^{(L)}(X)$, so $\|v(X)\| \le B$. The target is $Y = \langle w^\star, v(X)\rangle$ with $w^\star = \beta/\Theta$ and $\|w^\star\| \le 1$, and $|Y| \le B$. Fit squared loss over the unit Hilbert-space ball, to empirical objective tolerance $\varepsilon^2/8$.

Here is the statistical justification without dependence on path dimension. For any sample of size m, independent Rademacher signs ϵ_i give

$$\mathbb{E}_\epsilon \sup_{\|w\|\le 1} \frac{1}{m}\sum_i \epsilon_i\langle w, v_i\rangle = \frac{1}{m}\mathbb{E}_\epsilon\Big\|\sum_i \epsilon_i v_i\Big\| \le \frac{B}{\sqrt{m}}.$$

The inequality follows from Jensen's inequality and cancellation of cross terms. The squared loss lies in $[0, 4B^2]$ and is $4B$-Lipschitz in the prediction on $[-B, B]$. Symmetrization, the scalar contraction inequality applied to $(a - Y_i)^2 - Y_i^2$, and bounded differences therefore imply, with probability at least $1-\rho$,

$$\sup_{\|w\|\le 1} |R(w) - \widehat{R}(w)| \le \frac{32B^2}{\sqrt{m}} + 4B^2\sqrt{\frac{\log(2/\rho)}{2m}}.$$

For clarity, symmetrization bounds the expected supremum by twice the corresponding signed supremum; contraction bounds the latter by a universal constant times $4B$ times the linear signed supremum above. Including the sample-dependent constant Y_i^2 costs at most $B^2/\sqrt{m}$. Changing one sample changes the supremum by at most $4B^2/m$, giving the stated concentration term. The displayed constant 32 is conservative.

Choose

$$m_y = \left\lceil C\frac{B^4}{\varepsilon^4}\left(1 + \log\frac{4}{\delta}\right)\right\rceil$$

for a sufficiently large universal C so that the uniform deviation is at most $\varepsilon^2/8$, with $\rho = \delta/2$. The comparator $w^\star$ has both empirical and population loss zero. Hence the approximate empirical minimizer $\widehat{w}$ has population squared loss at most $\varepsilon^2/4$, and in particular at most $\varepsilon^2/2$.

The fit is computable from the scaled Gram matrix $H_{ij} = \Theta^2 K_L(X_i, X_j)$. Orthogonally projecting any feasible w onto the span of the training vectors preserves its predictions and cannot increase its norm. Thus it suffices to solve the finite convex problem

$$\min_{a\in\mathbb{R}^{m_y}} \frac{1}{m_y}\|Ha - y\|_2^2 \quad \text{subject to } a^\top Ha \le 1,$$

to the stated objective tolerance. Singular Gram matrices are allowed; no positive lower eigenvalue is assumed. Equivalently, $\widehat{\beta} = \Theta\widehat{w}$ has norm at most Θ and the resulting prediction is a linear combination of K_L. This is norm-constrained kernel least squares, not unconstrained interpolation.

Combining the errors. Deploy

$$\widehat{f}(x) = \mathrm{clip}_{[-B,B]}\langle \widehat{\beta}, \widehat{\Phi}^{(L)}(x)\rangle.$$

On the event of correct hidden signs, its squared error is at most the squared error of the fitted readout on the true feature, since clipping cannot increase distance to a target in $[-B, B]$. On the complementary event, clipping bounds the squared error by $4B^2$, even if the reconstructed feature has arbitrarily large norm. Equation 18 and the readout guarantee therefore yield

$$\mathbb{E}(\widehat{f}(X) - f^\star(X))^2 \le \varepsilon^2/2 + 4B^2\frac{\varepsilon^2}{8B^2} = \varepsilon^2.$$

Both training events hold with probability at least $1-\delta$. This proves the current real-valued statement without a margin at the readout.

Computation and scope. Maintaining all training scores costs $O(Mkm_g(d+M))$ operations for the gate fits; their prediction costs $O(Mk(d+M))$. Computing the readout Gram matrix costs $O(m_y^2(d+M))$, and its convex fit and prediction have polynomial cost in its size and requested tolerance. These bounds, $m_g + m_y$, and storage are polynomial in the parameters listed in the theorem, with no explicit path dimension. For rational data, products of binary-prefix inner products add only $O(\sum_r \log(n_r+1))$ bits. Dyadic normalization and gate score comparisons consequently have polynomial bit cost. Numerical encodings of Θ, B, γ, the training and query data, and optimization tolerances must additionally be charged in a bit-time implementation; no unit-cost claim for arbitrarily encoded numbers is intended. The assumptions are distribution-dependent. Bounded output values alone do not imply a bound on $\|\beta\|\|\Phi^{(L)}\|$, and small hidden-state margins do not automatically give path-space margins. The hardness result excludes a uniform polynomial guarantee for all the required parameters across its hard family; it does not identify which parameter must deteriorate for each target. □

H Additional details for the parity experiments

Teacher computation. Let $q^{(0)} = x$. At layer ℓ, adjacent values in $q^{(\ell-1)}$ are paired. For node r, the teacher computes

$$z_r^{(\ell)} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} q_{2r-1}^{(\ell-1)} \\ q_{2r}^{(\ell-1)} \end{bmatrix}, \qquad h_r^{(\ell)} = \mathrm{ReLU}(z_r^{(\ell)}), \tag{19}$$
$$q_r^{(\ell)} = \mathbf{1}^\top h_r^{(\ell)}, \qquad a_r^{(\ell)} = \mathbb{1}[z_r^{(\ell)} > 0].$$

The final scalar $q_1^{(L)}$ is the parity label. In the high-length experiment, the neural student replaces the fixed matrix above with one trainable matrix $W \in \mathbb{R}^{2\times 2}$ shared across every node and layer. Sharing is essential for length generalization: it allows the same learned local computation to be applied at depths beyond those observed in training.

Output-label loss. For a fitting minibatch B, the labels-only student minimizes

$$\mathcal{L}_{\mathrm{out}} = \frac{1}{|B|}\sum_{i\in B}\big(\widehat{f}_\theta(x_i) - y_i\big)^2. \tag{20}$$

The student output is continuous during training and is thresholded at $1/2$ for evaluation.

Binary-gate loss. Let $\widehat{z}_{iru}^{(\ell)}$ be a student preactivation and $a_{iru}^{(\ell)} \in \{0,1\}$ the corresponding teacher gate. With $K = 2(m-1)$ gates per example, we use

$$\mathcal{L}_{\mathrm{gate}} = \mathcal{L}_{\mathrm{out}} + \frac{\lambda}{|B|K}\sum_{i\in B}\sum_{\ell,r,u}\mathrm{BCE}\big(\mathrm{sigmoid}(\widehat{z}_{iru}^{(\ell)}), a_{iru}^{(\ell)}\big), \tag{21}$$
$$\mathrm{BCE}(p,a) = -a\log p - (1-a)\log(1-p).$$

Method	$m = 128$	$m = 256$	$m = 512$
Labels / Adam	> 2048	> 2048	> 2048
Gates / Adam	32	32	32
Guided activations / Adam	32	32	128
Ordered gates / LP	32	32	32

Table 1: Smallest tested training size for which every seed attains zero test error. The notation > 2048 means that the target was not reached within the tested grid.

Guided-activation loss. Ordinary post-ReLU matching can have zero gradient when the student is inactive but the teacher is active. We instead use an active-set-aware surrogate. During the auxiliary calculation, each shared student cell receives the corresponding teacher intermediate values as inputs. If $\widetilde{z}$ is its preactivation and (a, h) are the teacher gate and activation, define

$$\rho(\widetilde{z}; a, h) = a(\widetilde{z} - h)^2 + (1 - a)\,\mathrm{ReLU}(\widetilde{z})^2. \tag{22}$$

The complete objective is

$$\mathcal{L}_{\mathrm{guided}} = \mathcal{L}_{\mathrm{out}} + \frac{\lambda}{E_B} \sum_{i \in B} \sum_{\ell,r,u} \rho\big(\widetilde{z}_{iru}^{(\ell)}; a_{iru}^{(\ell)}, h_{iru}^{(\ell)}\big), \qquad E_B = \max\left\{ \sum_{i,\ell,r,u} (h_{iru}^{(\ell)})^2, 10^{-8} \right\}. \tag{23}$$

Only the auxiliary term is teacher-forced. The output loss is computed from the recursively evaluated student. We use $\lambda = 10$ for both auxiliary objectives.

Ordered-gate learner. The ordered learner pools revealed local states across nodes and layers and fits the two gates of the shared XOR cell as homogeneous halfspaces by linear programming. At test time, it reconstructs all gates and node outputs recursively from x and receives no teacher information.

Optimization and evaluation. Neural students are trained for 400 Adam updates with learning rate 0.015, cosine decay to 20% of the initial rate, minibatches of size at most 256, and global gradient norm clipped at 5. Each training set is split into an 80% fitting subset and a 20% validation subset; checkpoints are checked every 50 updates and selected by validation output MSE. The labels-only, gate-supervised, and guided-activation methods use three, three, and two random restarts, respectively. Because parameters are shared, one length-m trace supplies $2(m-1)$ local supervision targets; throughout, N denotes the number of top-level input strings, not the number of local targets.

For a test set of size M, classification error is

$$\widehat{R} = \frac{1}{M} \sum_{i=1}^{M} \mathbb{1}\Big[\mathbb{1}[\widehat{f}(x_i) \geq 1/2] \neq y_i\Big]. \tag{24}$$

Every reported mean and standard deviation is computed from five independent seed-level errors, with $20{,}000$ independent test inputs per seed.

High-length sample efficiency. Figure 2 reports the complete sample-efficiency results for $m \in \{128, 256, 512\}$, corresponding to depths $7, 8$, and 9. Labels-only learning remains near chance throughout. Gate supervision and the ordered-gate learner attain zero error for every tested setting. Guided activation supervision also attains zero error except at $m = 512, N = 32$, where its mean error is 19.86% with substantial between-seed variation; it reaches zero error for every seed by $N = 128$.

Sample attainment. Table 1 gives the smallest tested N for which all five seeds attain zero test error.

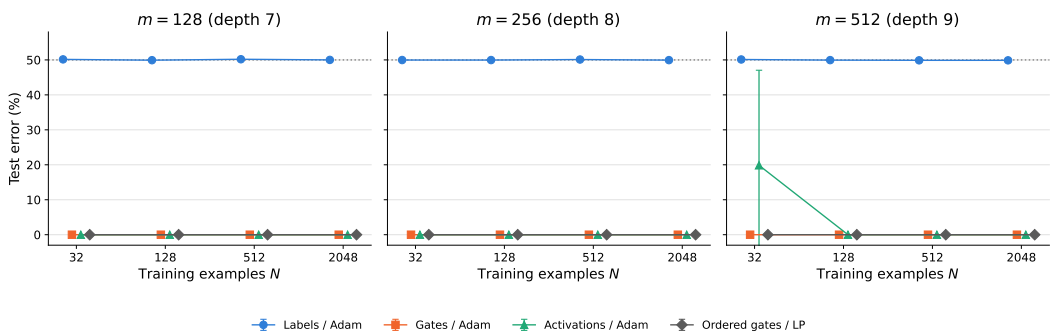


Figure 2: Test classification error for parity lengths $m \in \{128, 256, 512\}$. Points show means over five seeds, and error bars show between-seed sample standard deviations. The dotted line marks chance-level error. All methods receive only the input at test time.

m	N	Labels / Adam	Gates / Adam	Guided activations / Adam	Ordered gates / LP
128	32	50.15 ± 0.26	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
128	128	49.92 ± 0.33	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
128	512	50.19 ± 0.36	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
128	2048	50.00 ± 0.19	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
256	32	49.97 ± 0.33	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
256	128	49.96 ± 0.42	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
256	512	50.11 ± 0.19	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
256	2048	49.94 ± 0.13	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
512	32	50.12 ± 0.17	0.00 ± 0.00	19.86 ± 27.19	0.00 ± 0.00
512	128	49.94 ± 0.33	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
512	512	49.88 ± 0.45	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
512	2048	49.88 ± 0.54	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00

Table 2: Test classification error in percent (mean $\pm$ between-seed sample standard deviation). Lower is better.

Complete numerical results. Table 2 reports every high-length result.

Length generalization. For this experiment, models are trained only at $m_{\text{train}} = 512$ with $N = 2048$ and then evaluated without retraining at $m_{\text{test}} \in \{1024, 2048, 4096\}$. These lengths correspond to depths $10, 11$, and 12, all greater than the training depth 9. Figure 3 and Table 3 report test error at the training length and the three longer lengths. Gate supervision, guided activation supervision, and ordered-gate learning attain zero error for every length and seed, whereas labels-only error remains near 50%.

m_{test}	Labels	Gates	Activations	Ordered LP
512	49.88 ± 0.54	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
1024	49.94 ± 0.37	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
2048	49.96 ± 0.33	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
4096	49.89 ± 0.25	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00

Table 3: Length-generalization test error in percent after training at $m = 512$. Values are mean $\pm$ between-seed sample standard deviation.

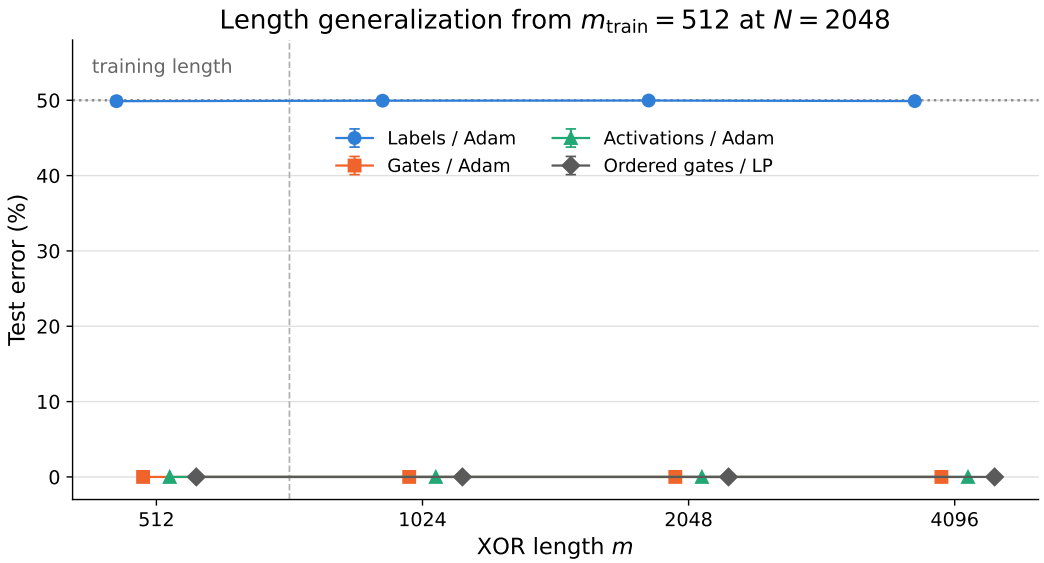


Figure 3: Length generalization after training at $m = 512$ with $N = 2048$. Points show means over five seeds, and error bars show between-seed sample standard deviations. The point at $m = 512$ uses held-out inputs at the training length; all remaining points are strictly beyond the training depth.