

Normal Alignment: Improved Cryptanalytic Sign Recovery on Hard-Label Networks

Shi Tang¹, Zirui Chen², Yongjia Su¹, Zhengchao Gao¹,
Lingyue Qin², and Xiaoyang Dong²

¹ Shandong University, Jinan, P. R. China
{shi.tang,yongjia.su,chao.qwq}@mail.sdu.edu.cn

² Tsinghua University, Beijing, P. R. China
chenzr25@mails.tsinghua.edu.cn
{qinly,xiaoyangdong}@tsinghua.edu.cn

Abstract. At EUROCRYPT 2025, Carlini *et al.* proposed a breakthrough in the cryptanalytic extraction on hard-label (S1) deep neural networks (DNNs), demonstrating polynomial-time signature and sign recovery. However, Carlini *et al.*’s sign-recovery method (which we call *Future Toggle*) suffers only a marginal advantage over random guessing, producing high-confidence wrong sign predictions in deeper layers. Such errors trigger expensive exponential-time enumeration.

This work presents *Normal Alignment*, a novel statistical sign-recovery approach for S1 DNNs. Drawing on the expected length difference between projected normals of adjacent decision facets at dual points, our method infers neuron signs via normal-signature alignment. It delivers higher voting accuracy and pushes erroneous predictions to low-confidence ranks, which further enables a more efficient combined method, *eSOE + Alignment*, by combining *Normal Alignment* with the hard-label SOE extension. This combined strategy removes heavy enumeration overhead and realizes exact polynomial-time full sign recovery.

Experiments demonstrate the effectiveness of our method, especially for deep layers. For example, with our method, the signs for CIFAR-10 (architecture 192-64×8-10) and MNIST (architecture 64-96×3-32-10) models can be fully recovered in polynomial time; in contrast, Carlini *et al.*’s sign-recovery method would require exponential-time enumerations involving 2^{52} or 2^{82} guesses of the signs, respectively.

Keywords: Cryptanalytic model extraction · ReLU networks · Sign recovery · S1 access · Normal Alignment

1 Introduction

Deep neural networks (DNNs) are widely used in computer vision [16], natural-language processing [24], and medical diagnosis [11], etc. Training high-performing DNNs often requires large amounts of data, computation and engineering efforts, making trained models valuable intellectual assets [20]. Model extraction is a long-studied attack in which an adversary uses input-output queries of the victim

DNN (or other side-channel information [2]) to extract its parameters (weights and biases). Early works explored network reconstruction [12] and query-based model stealing [19,23]. At CRYPTO 2020, Carlini, Jagielski, and Mironov introduced a cryptanalytic approach to model extraction [6], which exploits the piecewise-affine structure of a ReLU network and recovers its parameters layer by layer from raw-output queries. Layer-wise extraction proceeds in two stages: First, the *signature recovery* identifies the unsigned weights and biases of each neuron by using the high-order differential at a so-called *critical point*, where one ReLU input of the DNN is exactly zero; Second, the *sign recovery* determines the sign. Their method enjoys a polynomial-time *signature recovery* phase, but suffers from an exponential-time *sign recovery* phase by a brute-force guessing method. At EUROCRYPT 2024, Canales-Martínez *et al.* [4] developed polynomial-time *sign recovery* algorithms in raw-output setting, *i.e.*, the *Neuron Wiggle* and *SOE* methods. Later work examined the practical limitations of these layer-wise attacks. Foerster *et al.* [13] found that increasing the number of critical points does not necessarily improve *Neuron Wiggle* recovery for difficult neurons. Liu *et al.* [18] addressed rank-deficient signature systems and the misattribution of critical points from deeper layers, thereby extending practical extraction from three hidden layers to eight. In parallel, cryptanalytic extraction has expanded along two dimensions: covering various activation functions [8,21,1] and different network architectures [25,22,17,9].

Hard-label extraction. The S1 hard-label setting returns only the predicted class labels (*e.g.*, “dog” or “car”) and hides the logits. At ASIACRYPT 2024, Yi Chen *et al.* initiated the cryptanalytic extraction in the S1 setting [7], but it requires exponential execution time. At EUROCRYPT 2025, Carlini *et al.* [5] gave the first polynomial-query, polynomial-time hard-label extraction. The attack collects and clusters *dual points*, which are critical and also on a visible class-decision boundary, to recover the signatures. At CRYPTO 2026, Ito, Miura, and Todo [15] identified a limitation of Carlini *et al.*’s attack [5]: for nearly always-active neurons, the state switches needed for parameter recovery can become exponentially difficult to observe. They proposed cross-layer extraction to address this failure mode. In 2026, Zirui Chen *et al.* [10] proposed the *Approximate Signature Vector* (ASV) method to reduce the cost of clustering dual points, and hence improve Carlini *et al.*’s signature recovery phase [5].

Existing Sign Recovery Methods in S1 and Their Limitations. There are only two existing S1 *sign recovery* methods: Carlini *et al.*’s statistical *Future Toggle* method [5], and Canales-Martínez and Santos’s deterministic hard-label *SOE* method [3]. The hard-label *SOE* typically recovers only the first hidden layer, unless the network is sufficiently contractive to permit the recovery of deeper layers. The main limitation of *Future Toggle* is its *weak advantage over random guessing*. In our practical experiments, *Future toggle* suffers small advantage than random guessing, leading to the low vote accuracy and the low vote confidence. For example, on the evaluated CIFAR-10 model, Table 2 re-

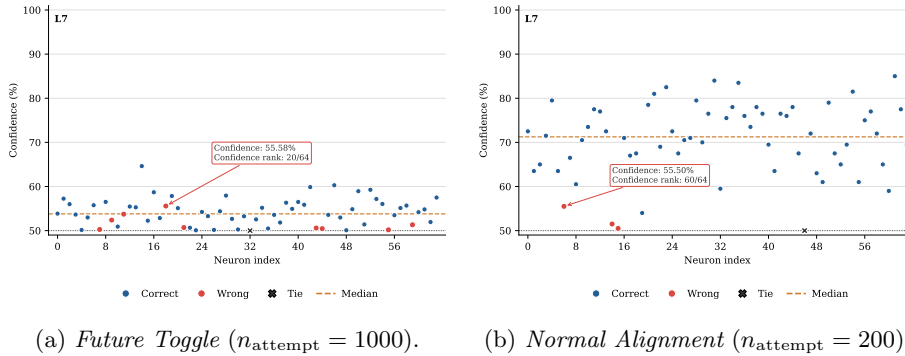


Fig. 1: vote confidence distributions of *Future Toggle* and *Normal Alignment* on the 7th hidden layer of the CIFAR-10 model with architecture $192\text{-}64\times 8\text{-}10$.

ports vote accuracies of 52%–57%, while Figure 1a shows that the confidence of most neurons is below 60%.

The weak advantage may lead to many incorrectly recovered signs in deep layers, which results in an exponential time complexity to recover the full signs, shifting the entire hard-label extraction from polynomial to exponential time complexity. Specifically, the *Future Toggle* may produce errors with relatively high confidence. As shown in Figure 1a, 9 of the 64 signs on layer 7 are incorrectly recovered, including one with a confidence of 55.58% that ranks 20th among all neurons by confidence. This makes exact recovery of the entire layer difficult under two existing approaches.

- **Enumeration Infeasible.** To recover all signs, we follow the approach of Foerster et al. [13]: we enumerate and guess the low-confidence sign assignments (which may contain errors) until all erroneous signs are covered, and verify the assignments by executing the next-layer signature recovery algorithm. As shown in Figure 1a, since the erroneous sign ranks 20-th in confidence, the enumeration must cover all 45 signs from rank 20 to rank 64. This requires testing 2^{45} possible assignments.
- **SOE Failure.** In 2026, Liu et al. [18] combined statistical predictions (*Neuron Wiggle*) with *SOE* in raw-output setting by eliminating unknowns associated with neurons predicted to be inactive with high confidence. Hence, when an actually active neuron is incorrectly predicted to be inactive with high confidence, its nonzero unknown is likely to be eliminated, making the reduced *SOE* incorrect and causing the combined strategy to fail.

Our Contributions. This paper introduces a new sign recovery method in S1 setting, called *Normal Alignment*, which uses the normals of the two decision facets adjacent to a dual point to directly determine the target sign, without repeatedly walking along the decision boundary to search for neuron toggles in

future layers (*i.e.*, Carlini *et al.*'s *Future Toggle* method [5]). Our method is based on the following statistical intuition (which is also formally proved): at a dual point, the decision-facet normal on the active side has a larger expected length than the normal on the inactive side, because it explicitly includes the target neuron's weight contribution. The advantages of our methods are summarized below:

1. **Larger Statistical Advantage.** As shown in Table 7, with the same budget of $n_{\text{attempt}} = 200$ dual points, *Normal Alignment* achieves an overall vote accuracy of 68.74% across all hidden layers, compared with 60.89% for *Future Toggle*. In deep hidden layers in Table 6, our method still maintains a vote accuracy of about 70%, while the vote accuracy of *Future Toggle* drops to around 53% in layers 3-7.
2. **Fewer Queries Needed.** *Normal Alignment* neither walks along the decision boundary, nor discards dual points due to non-future neuron toggles – as occurs with *Future Toggle*. When the two adjacent decision-facet normals are successfully recovered, a dual point can produce a vote. Consequently, the dual point utilization reaches 100%, as shown in Table 6. By contrast, only 2.26% of the dual points produce a vote in Layer 8 by the *Future Toggle*. Furthermore, the weak advantage of the *Future Toggle* naturally requires more votes to improve accuracy, and hence usually needs more queries and time than *Normal Alignment* as shown in Table 1.
3. **Low Confidence for Incorrect Signs.** As shown in Figure 1, For *Normal Alignment*, the highest-confidence error has a confidence of 55.50% and ranks 60th among the 64 neurons, whereas *Future Toggle*'s highest-confidence error has a similar confidence of 55.58% but ranks as high as 20th. Therefore, to recover the full signs by enumeration [13], our method must test 2^5 sign assignments, while *Future Toggle* tests 2^{45} sign assignments. We also test more models with different layers in Table 5 in Supp. A. Across all hidden layers of each model, our highest-ranked errors occur on CIFAR-10 L6 and MNIST L2 at ranks 50/64 and 68/96, respectively, yielding enumeration complexities of 2^{15} and 2^{29} . By contrast, at the same budget of $n_{\text{attempt}} = 200$, the highest-ranked errors of *Future Toggle* require enumeration complexities of 2^{64} and 2^{93} on the CIFAR-10 and MNIST models, respectively.
4. **Feasible Combination of the Hard-label SOE and Normal Alignment: eSOE + Alignment.** Because the highest-confidence error ranks very low in *Normal Alignment*, a similar combination of a statistical method and a deterministic method by Liu *et al.* [18] in S5 setting works in S1 setting, *i.e.*, combining hard-label SOE and *Normal Alignment*. The *Normal Alignment* identifies the high-confidence (highly ranked) inactive neurons and eliminates the corresponding zero equations in SOE. By contrast, *Future Toggle* is susceptible to high-confidence errors (*i.e.*, errors with high confidence rank); for instance, an active neuron might be predicted as inactive with high confidence, causing a nonzero equation to be erroneously discarded and thus causing the SOE method to fail. Similarly to Liu *et al.* [18], to further increase the rank of SOE, we select several transition points

sharing the same activation states in the target and future layers, forming a stacked coefficient matrix. Besides, we also introduce an orthogonal projection matrix to eliminate the unknown normal length at each point. Therefore, we call the resulting method hard-label *SOE* extension and *Normal Alignment* (*eSOE + Alignment*). It helps eliminate the enumeration complexities – specifically the 2^{15} time for layer L6 of the CIFAR-10 network (Table 6) and the 2^{29} time for layer L2 of the MNIST network (Table 7).

Experiments. As shown in Table 1, the sign recovery methods are evaluated on CIFAR-10 and MNIST networks. We follow the same assumption as [4]: when targeting the layer k , the preceding layers ($< k$) and the unsigned signatures of layer k are known. Tables 6 and 7 in Supp. A report the full results of our experiments, comparing *Normal Alignment*, *eSOE + Alignment*, *Future Toggle*, the combination of the hard-label *SOE* extension and *Future Toggle* (*eSOE + Toggle*) for a fair comparison though *eSOE + Toggle* does not reduce the overall enumeration complexities as shown in Table 1. Our *eSOE+Alignment* correctly recovers all 512/512 signs on CIFAR-10 and all 320/320 signs on MNIST, thereby achieving exact sign recovery in polynomial time in Table 1. In contrast, *eSOE+Toggle* recovers only 469/512 and 296/320 signs, respectively. Specifically, for the CIFAR-10 model with $n_{\text{attempt}} = 1000$ in Table 6, *eSOE + Toggle* leaves errors on L5, L7, and L8, whose highest-confidence erroneous signs rank 44-th, 20-th, and 13-th out of 64, respectively. Exact recovery must therefore enumerate the 52 signs from rank 13 to rank 64 in L8, requiring 2^{52} sign enumerations. For the MNIST model in Table 7, the highest-confidence error ranks 15-th among 96 neurons on L3, hence requiring 2^{82} sign enumerations to recover full signs. The source code for all the experiments can be found via

XXX

2 Preliminaries

Unless otherwise specified, the subscript and superscript numbers start from 1.

- $[m]$: for a positive integer m , we write $[m] = \{1, \dots, m\}$,
- $\mathbf{A}$: matrix, where its i -th row is $\mathbf{A}_i$, and its element in i -th row and j -th column is $\mathbf{A}_{i,j}$, $i, j \geq 1$,
- $\mathbf{x}$: column vector, and its i -th element is $\mathbf{x}_i$, $i \geq 1$,
- $\mathcal{F}$: functions,
- $\mathbb{C}, \mathbb{D}$: space or set,
- neuron (k, j) : the j -th neuron in layer k .

2.1 Notations and Definitions

The DNN is composed of a sequence of functions alternating between linear functions $f^{(k)} : \mathbb{R}^{d^{(k)}} \mapsto \mathbb{R}^{d^{(k+1)}}$ ($k \in [r+1]$), and a nonlinear function σ (component-wise ReLU function):

$$\mathcal{F}_\theta = f^{(r+1)} \circ \sigma \circ f^{(r)} \circ \sigma \circ \dots \circ f^{(2)} \circ \sigma \circ f^{(1)}, \quad (1)$$

Table 1: Comparison of our *eSOE+Alignment* and *eSOE+Toggle* [5] on the CIFAR-10 and MNIST models.

Model and Method			Recovery Complexity		Experimental Results		
Architecture	Method	n_{attempt}	Method execution	Complete recovery	Correct signs	Time	Queries
CIFAR-10 192-64×8-10	eSOE+Alignment	200	<i>poly</i>	<i>poly</i>	512/512	$2^{10.73}$	$2^{31.13}$
	eSOE+Toggle [5]	1000	<i>poly</i>	<i>exp</i>	469/512	$2^{13.47}(s) + 2^{52}(g)$	$2^{35.49}$
MNIST 64-96×3-32-10	eSOE+Alignment	200	<i>poly</i>	<i>poly</i>	320/320	$2^{11.20}$	$2^{28.71}$
	eSOE+Toggle [5]	1000	<i>poly</i>	<i>exp</i>	296/320	$2^{17.39}(s) + 2^{82}(g)$	$2^{30.77}$

n_{attempt} : denotes the number of attempted dual points per neuron. As shown in Fig. 3, we set $n_{\text{attempt}} = 200$ for *Normal Alignment*. Because *Future Toggle* provides a weaker statistical advantage, we set $n_{\text{attempt}} = 1000$ for *Future Toggle*.

Method execution: denotes the time complexity of *eSOE+Alignment* or *eSOE+Toggle*. Since hard-label *SOE*, *Normal Alignment*, and *Future Toggle* all run in polynomial time, both combined methods also run in polynomial time.

Complete recovery: denotes the time complexity required to recover all neuron signs correctly. *eSOE+Alignment* recovers all neuron signs correctly and therefore achieves complete recovery in polynomial time. In contrast, *eSOE+Toggle* leaves some signs incorrect; guaranteeing complete recovery therefore requires exponential enumeration of the unresolved sign assignments [13].

Time: consists of two components. The notation 2^x (s) denotes the time of *eSOE+Alignment/Toggle* in seconds, whereas 2^y (g) denotes the cost of sign guessing required for complete recovery of all signs in the model.

Queries: In the CIFAR-10 proof-of-concept experiments, the decision-facet normals are computed directly from model parameters, and the reported values estimate the corresponding hard-label query cost; the MNIST entries report the actual hard-label query counts.

where $f^{(k)} : \mathbb{R}^{d^{(k)}} \rightarrow \mathbb{R}^{d^{(k+1)}}$ is an affine transformation:

$$\mathbf{y}^{(k)} = f^{(k)}(\mathbf{x}^{(k)}) = \mathbf{A}^{(k)}\mathbf{x}^{(k)} + \mathbf{b}^{(k)} \in \mathbb{R}^{d^{(k+1)}}, \quad (2)$$

where $\mathbf{x}^{(k)} \in \mathbb{R}^{d^{(k)}}$ represents the input vector of layer k , and $\mathbf{x}^{(1)} \in \mathbb{R}^{d^{(1)}}$ is the model input. The weight matrix $\mathbf{A}^{(k)} \in \mathbb{R}^{d^{(k+1)} \times d^{(k)}}$ and the bias vector $\mathbf{b}^{(k)} \in \mathbb{R}^{d^{(k+1)}}$ are composed of floating-point numbers, which are the model parameters.

Given input $\mathbf{x}^{(1)} \in \mathbb{R}^{d^{(1)}}$, the ReLU function σ in layer k is also interpreted as the matrix determined by $\mathbf{x}^{(1)}$,

$$\mathbf{I}^{(k)} = \text{diag}(\tau_1^{(k)}, \tau_2^{(k)}, \dots, \tau_{d^{(k+1)}}^{(k)}), \quad (3)$$

where $\tau_i^{(k)} = 1$, $i \in [d^{(k+1)}]$ when $\mathbf{y}_i^{(k)} \geq 0$, else $\tau_i^{(k)} = 0$. Then, $\mathbf{x}^{(k+1)} = \mathbf{I}^{(k)}\mathbf{y}^{(k)} \in \mathbb{R}^{d^{(k+1)}}$ is the output of layer k .

Definition 1 (Linear Neighborhood). Given an input $\mathbf{x} \in \mathbb{R}^{d^{(1)}}$, the matrices $\mathbf{A}^{(k)}$, $\mathbf{b}^{(k)}$, $\mathbf{I}^{(k)}$, $k \in [r+1]$ will be all fixed. The linear neighborhood of $\mathbf{x}$ is defined as the subset $\mathbb{L}_{\mathbf{x}} \subset \mathbb{R}^{d^{(1)}}$, so that, for all $\mathbf{x}^{(1)} \in \mathbb{L}_{\mathbf{x}}$, the same matrices $\mathbf{A}^{(k)}$, $\mathbf{b}^{(k)}$, $\mathbf{I}^{(k)}$ will be applied to compute the output of the DNN.

The DNN has been proved to be a piecewise linear function [6,4], i.e., for $\mathbf{x} \in \mathbb{L}_{\mathbf{x}}$, the model output will change linearly, i.e., the DNN is reduced to

$$\begin{aligned} \mathcal{F}_{\theta}(\mathbf{x}) &= \mathbf{A}^{(r+1)} \left(\mathbf{I}^{(r)} \left(\mathbf{A}^{(r)} \dots \left(\mathbf{I}^{(1)} \left(\mathbf{A}^{(1)}\mathbf{x} + \mathbf{b}^{(1)} \right) \right) \dots + \mathbf{b}^{(r)} \right) \right) + \mathbf{b}^{(r+1)} \\ &= \mathbf{A}^{(r+1)} \mathbf{I}^{(r)} \mathbf{A}^{(r)} \dots \mathbf{I}^{(2)} \mathbf{A}^{(2)} \mathbf{I}^{(1)} \mathbf{A}^{(1)} \mathbf{x} + \boldsymbol{\beta} = \boldsymbol{\Gamma} \mathbf{x} + \boldsymbol{\beta}. \end{aligned} \quad (4)$$

Definition 2 (Oracle models). In S5 raw-output setting, the oracle returns all logits: $\mathcal{O}_{S5}(\mathbf{x}) = \mathcal{F}_\theta(\mathbf{x})$. In S1 hard-label setting, it returns only the label

$$\mathcal{O}_{S1}(\mathbf{x}) = \min \arg \max_{j \in [d^{(r+2)}]} \mathcal{F}_\theta(\mathbf{x}),$$

where the minimum indicates a deterministic rule for ties.

Definition 3 (Critical hyperplane, activation boundary, and critical point). The critical hyperplane of neuron (k, j) (the j -th neuron of layer k) in the input space of layer k is

$$\mathbb{H}_j^{(k)} = \{\mathbf{u} \in \mathbb{R}^{d^{(k)}} : \langle \mathbf{A}_j^{(k)}, \mathbf{u} \rangle + \mathbf{b}_j^{(k)} = 0\}, k \in [r+1], j \in [d^{(k+1)}].$$

The corresponding activation boundary in the model-input space is

$$\mathbb{C}_j^{(k)} = \{\mathbf{x}^{(1)} \in \mathbb{R}^{d^{(1)}} : \mathbf{x}^{(k)} \in \mathbb{H}_j^{(k)}\}.$$

Therefore, $\mathbf{x}^{(1)} \in \mathbb{C}_j^{(k)}$ is a critical point of the neuron (k, j) .

Prefix and suffix maps of layer k . Fix a target layer $k \in [r+1]$. We decompose the network as

$$\mathcal{F}_\theta = \underbrace{f^{(r+1)} \circ \sigma \circ f^{(r)} \circ \dots \circ \sigma \circ f^{(k+1)}}_{\mathcal{G}_x^{k+1}} \circ \sigma \circ f^{(k)} \circ \underbrace{\sigma \circ f^{(k-1)} \circ \dots \circ \sigma \circ f^{(1)}}_{\mathcal{F}_x^{k-1}}, \quad (5)$$

where $\mathcal{F}_x^{k-1}$ is the layers before layer k , and $\mathcal{G}_x^{k+1}$ is the layers after the layer k . Given model input $\mathbf{x}$ and its corresponding linear neighborhood $\mathbb{L}_\mathbf{x}$, the $\mathcal{F}_x^{k-1}$ and $\mathcal{G}_x^{k+1}$ collapse to fix affine functions, i.e., for model input $\mathbf{x}^{(1)} \in \mathbb{L}_\mathbf{x}$, and its corresponding $\mathbf{x}^{(k)}$,

$$\mathbf{x}^{(k)} = \mathcal{F}_x^{k-1}(\mathbf{x}^{(1)}) = \mathbf{F}^{(k-1)} \mathbf{x}^{(1)} + \boldsymbol{\beta}^{(k-1)}, \quad \mathcal{G}_x^{k+1}(\mathbf{x}^{(k+1)}) = \mathbf{G}^{(k+1)} \mathbf{x}^{(k+1)} + \boldsymbol{\gamma}^{(k+1)}. \quad (6)$$

where $\mathbf{F}^{(k-1)} = \mathbf{I}^{(k-1)} \mathbf{A}^{(k-1)} \dots \mathbf{I}^{(1)} \mathbf{A}^{(1)}$, $\mathbf{G}^{(k+1)} = \mathbf{A}^{(r+1)} \mathbf{I}^{(r)} \mathbf{A}^{(r)} \dots \mathbf{I}^{(k+1)} \mathbf{A}^{(k+1)}$. according to Eq. (4), and hence Eq. (4) becomes

$$\mathcal{F}_\theta(\mathbf{x}^{(1)}) = \mathbf{G}^{(k+1)} \mathbf{I}^{(k)} \left(\mathbf{A}^{(k)} \left(\mathbf{F}^{(k-1)} \mathbf{x}^{(1)} + \boldsymbol{\beta}^{(k-1)} \right) + \mathbf{b}^{(k)} \right) + \boldsymbol{\gamma}^{(k+1)}. \quad (7)$$

Definition 4 (Transition and dual points, decision boundary, decision facet). For distinct classes $a, b \in [d^{(r+2)}]$, define $\mathcal{D}_{ab}(\mathbf{x}) = \mathcal{F}_\theta(\mathbf{x})_a - \mathcal{F}_\theta(\mathbf{x})_b$, and the set of decision boundary

$$\mathbb{D}_{ab} = \{\mathbf{x} \in \mathbb{R}^{d^{(1)}} : \mathcal{F}_\theta(\mathbf{x})_a = \mathcal{F}_\theta(\mathbf{x})_b > \mathcal{F}_\theta(\mathbf{x})_c \text{ for every } c \in [d^{(r+2)}] \setminus \{a, b\}\}. \quad (8)$$

Any $\mathbf{x} \in \mathbb{D}_{ab}$ is a transition point for switching classes a and b , whose decision facet in the input model space $\mathbb{R}^{d^{(1)}}$ is defined as $\mathbb{D}_{ab} \cap \mathbb{L}_\mathbf{x} \in \mathbb{R}^{d^{(1)}}$. The point $\mathbf{x} \in \mathbb{D}_{ab} \cap \mathbb{C}_j^{(k)}$ is a dual point for neuron (k, j) and class pair (a, b) . Usually, there are at least two adjacent decision facets for a given dual point $\mathbf{x}$, denoted as $\mathbb{D}_{ab} \cap \mathbb{L}_\mathbf{x}$ and $\mathbb{D}_{ab} \cap \mathbb{L}'_\mathbf{x}$.

Figure 2 summarizes different types of point in geometry, where each cell represents a linear neighborhood.

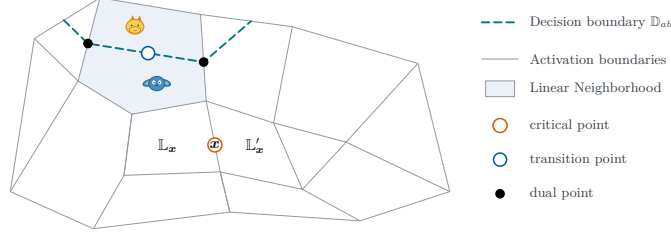


Fig. 2: Piecewise-affine geometry in the model-input space.

Definition 5 (Layer- k wiggle). A layer- k wiggle around a model input $\mathbf{x}$ is a small vector $\delta^{(k)} \in \mathbb{R}^{d^{(k)}}$, so that there exists $\delta^{(1)} \in \mathbb{R}^{d^{(1)}}$ that satisfies $\mathbf{x} + \delta^{(1)} \in \mathbb{L}_{\mathbf{x}}$ (or $\mathbf{x} - \delta^{(1)} \in \mathbb{L}_{\mathbf{x}}$) and $\delta^{(k)} = \mathbf{F}^{(k-1)}(\mathbf{x} + \delta^{(1)}) + \beta^{(k-1)} - (\mathbf{F}^{(k-1)}(\mathbf{x}) + \beta^{(k-1)}) = \mathbf{F}^{(k-1)}(\delta^{(1)})$ (or $\delta^{(k)} = \mathbf{F}^{(k-1)}(\mathbf{x}) + \beta^{(k-1)} - (\mathbf{F}^{(k-1)}(\mathbf{x} - \delta^{(1)}) + \beta^{(k-1)}) = \mathbf{F}^{(k-1)}(\delta^{(1)})$).

Definition 6 (Control space). Given a model input $\mathbf{x}$ and its linear neighborhood $\mathbb{L}_{\mathbf{x}}$, its control space at the input to layer k is

$$\mathbb{V}_{\mathbf{x}}^{(k)} := \text{span} \left\{ \delta^{(k)} = \mathbf{F}^{(k-1)} \delta^{(1)} : \delta^{(1)} \in \mathbb{R}^{d^{(1)}}, \mathbf{x} + \delta^{(1)} \in \mathbb{L}_{\mathbf{x}} \right\} \subseteq \mathbb{R}^{d^{(k)}}.$$

Equivalently, $\mathbb{V}_{\mathbf{x}}^{(k)}$ is the subspace spanned by the columns of $\mathbf{F}^{(k-1)}$.

Definition 7 (Projection onto the control space). Since the control space $\mathbb{V}_{\mathbf{x}}^{(k)}$ is the column space of $\mathbf{F}^{(k-1)}$, its orthogonal complement satisfies $(\mathbb{V}_{\mathbf{x}}^{(k)})^{\perp} = \ker \left((\mathbf{F}^{(k-1)})^{\top} \right)$. Every vector $\mathbf{u} \in \mathbb{R}^{d^{(k)}}$ can therefore be uniquely written as $\mathbf{u} = \bar{\mathbf{u}} + \mathbf{u}_{\perp}$, where $\bar{\mathbf{u}} \in \mathbb{V}_{\mathbf{x}}^{(k)}$ and $\mathbf{u}_{\perp} \in (\mathbb{V}_{\mathbf{x}}^{(k)})^{\perp}$. We call $\bar{\mathbf{u}}$ the orthogonal projection of $\mathbf{u}$ onto $\mathbb{V}_{\mathbf{x}}^{(k)}$ and write $\bar{\mathbf{u}} := \mathbf{P}^{(k)} \mathbf{u}$, where $\mathbf{P}^{(k)}$ is the corresponding orthogonal projection matrix. Equivalently, if the columns of $\mathbf{Q}^{(k)}$ form an orthonormal basis of $\mathbb{V}_{\mathbf{x}}^{(k)}$, then

$$\mathbf{P}^{(k)} = \mathbf{Q}^{(k)} (\mathbf{Q}^{(k)})^{\top}, \quad \bar{\mathbf{u}} = \mathbf{Q}^{(k)} (\mathbf{Q}^{(k)})^{\top} \mathbf{u}.$$

In particular, $\mathbf{P}^{(k)}$ is symmetric, i.e., $(\mathbf{P}^{(k)})^{\top} = \mathbf{P}^{(k)}$.

2.2 Extraction Goal and Assumptions

Definition 8 (Signature [4]). Let $\mathbf{A}_j^{(k)} = (\mathbf{A}_{j,1}^{(k)}, \dots, \mathbf{A}_{j,d^{(k)}}^{(k)})$ be the weight vector of neuron (k, j) , and assume that $\mathbf{A}_{j,1}^{(k)} \neq 0$. Its signature is the vector

$$\hat{\mathbf{A}}_j^{(k)} := \frac{\mathbf{A}_j^{(k)}}{\mathbf{A}_{j,1}^{(k)}} = \left(1, \frac{\mathbf{A}_{j,2}^{(k)}}{\mathbf{A}_{j,1}^{(k)}}, \dots, \frac{\mathbf{A}_{j,d^{(k)}}^{(k)}}{\mathbf{A}_{j,1}^{(k)}} \right), \quad (9)$$

where the true weight vector satisfies $\mathbf{A}_j^{(k)} = \mathbf{A}_{j,1}^{(k)} \hat{\mathbf{A}}_j^{(k)}$.

Thus, after signature recovery, the only remaining ambiguity is the nonzero scalar $\mathbf{A}_{j,1}^{(k)}$. Its magnitude does not need to be recovered [6]: since $\text{ReLU}(cz) = c \text{ReLU}(z)$ for every $c > 0$, a positive scaling can be absorbed into the outgoing weights of the neuron. Its sign, however, is essential. Negating the recovered affine form exchanges its active and inactive sides and cannot be absorbed through ReLU. We call the sign of $\mathbf{A}_{j,1}^{(k)}$ the sign of neuron (k, j) . Sign recovery determines this sign and thereby identifies the true active side of the neuron. Denote signs in the layer k by $(s_1, s_2, \dots, s_{d^{(k+1)}})$, $s_j \in \{1, -1\}, \forall j \in [d^{(k+1)}]$, and define the sign matrix

$$\mathbf{S}^{(k)} = \text{diag}(s_1^{(k)}, s_2^{(k)}, \dots, s_{d^{(k+1)}}^{(k)}). \quad (10)$$

Then, given $\mathbf{x} \in \mathbb{R}^{d^{(1)}}$, by Eq. (7), the model output in S5 setting is

$$\mathcal{F}_\theta(\mathbf{x}) = \mathbf{G}^{(k+1)} \mathbf{I}^{(k)} \mathbf{S}^{(k)} \left(\hat{\mathbf{A}}^{(k)} \left(\mathbf{F}^{(k-1)} \mathbf{x} + \boldsymbol{\beta}^{(k-1)} \right) + \hat{\mathbf{b}}^{(k)} \right) + \boldsymbol{\gamma}^{(k+1)}, \quad (11)$$

where $\hat{\mathbf{A}}^{(k)}$ and $\hat{\mathbf{b}}^{(k)}$ are the unsigned signatures and biases in layer k .

Extraction Goal. Our final objective is functionally equivalent parameter extraction: given oracle access to a target network $\mathcal{F}_\theta$, recover parameters $\hat{\theta}$ such that the extracted network computes the same function as the target, up to unavoidable symmetries such as positive neuron rescaling and permutation within a layer. This paper focuses on sign recovery in S1 hard-label setting and assumes that a preceding signature-recovery phase has recovered the target signatures up to nonzero scalar multiples.

Assumptions.

- **Known architecture.** The attacker knows $(d^{(1)}, \dots, d^{(r+2)})$ and that the hidden layers are fully connected ReLU layers.
- **Full-domain inputs.** The attacker may adaptively query any input in $\mathbb{R}^{d^{(1)}}$.
- **Precise computation.** The analysis assumes exact real arithmetic or sufficiently high floating-point precision.
- **Oracle access.** We consider both S5 raw-output and S1 hard-label access.

- **Available signatures.** For the target layer k , we assume that all the weights and biases of the preceding layers $1, \dots, k-1$ are recovered, while each neuron’s signature $\hat{\mathbf{A}}^{(k)}$ and bias $\hat{\mathbf{b}}^{(k)}$ are known up to an unknown nonzero scalar in layer k . Also, we assume that no two signatures are the same [4]. Our goal is to recover the signs in layer k .

3 Existing Sign Recovery Methods in S1 Setting and their Limitations

3.1 Future Toggle in Hard-label Setting

Neuron Wiggle in S5 Setting [4]. The *Neuron Wiggle* method, proposed by Canales-Martínez *et al.* at EUROCRYPT 2024, is a heuristic sign recovery method in raw-output setting. The method relies on a basic asymmetry across the target activation boundary: *the norm of the target layer output change is larger on the target active side because ReLU blocks the target neuron’s contribution on the inactive side.* We first establish this asymmetry and then explain how it motivates the observation used by *Future Toggle* [5] in S1 setting.

Suppose that $\mathbf{x}$ is a critical point of the target neuron (k, j) , and denote its two adjacent linear neighborhoods by $\mathbb{L}_{\mathbf{x}}$ and $\mathbb{L}'_{\mathbf{x}}$. Their activation statuses differ only in the state of neuron (k, j) . Without loss of generality, suppose that the target neuron is active in $\mathbb{L}_{\mathbf{x}}$ and inactive in $\mathbb{L}'_{\mathbf{x}}$. Denote the corresponding activation matrices in layer k by $\mathbf{I}_+^{(k)}$ and $\mathbf{I}_-^{(k)}$. Choose a wiggle $\boldsymbol{\delta}^{(1)} \in \mathbb{R}^{d^{(1)}}$ such that $\mathbf{x} + \boldsymbol{\delta}^{(1)} \in \mathbb{L}_{\mathbf{x}}$ and $\mathbf{x} - \boldsymbol{\delta}^{(1)} \in \mathbb{L}'_{\mathbf{x}}$. It induces the layer- k wiggle $\boldsymbol{\delta}^{(k)} := \mathbf{F}^{(k-1)} \boldsymbol{\delta}^{(1)}$ by Def. 5. Hence, the layer- $(k+1)$ wiggles are $\boldsymbol{\delta}^{(k+1)} = \mathbf{I}_+^{(k)} \mathbf{A}^{(k)} \boldsymbol{\delta}^{(k)}$ and $\boldsymbol{\delta}'^{(k+1)} = \mathbf{I}_-^{(k)} \mathbf{A}^{(k)} \boldsymbol{\delta}^{(k)}$ on the active and inactive sides, respectively. Let $\mathbb{S}$ denote the set of active neurons in layer k within $\mathbb{L}_{\mathbf{x}}$. We have

$$\|\boldsymbol{\delta}^{(k+1)}\|^2 = \sum_{i \in \mathbb{S}} |\mathbf{A}_i^{(k)} \boldsymbol{\delta}^{(k)}|^2, \quad \|\boldsymbol{\delta}'^{(k+1)}\|^2 = \sum_{i \in \mathbb{S} \setminus \{j\}} |\mathbf{A}_i^{(k)} \boldsymbol{\delta}^{(k)}|^2. \quad (12)$$

Consequently,

$$\|\boldsymbol{\delta}^{(k+1)}\|^2 = \|\boldsymbol{\delta}'^{(k+1)}\|^2 + |\mathbf{A}_j^{(k)} \boldsymbol{\delta}^{(k)}|^2. \quad (13)$$

Therefore, whenever $\mathbf{A}_j^{(k)} \boldsymbol{\delta}^{(k)} \neq 0$, the layer- $(k+1)$ wiggle has a strictly larger norm on the active side. For one coordinate of the model output vector (*e.g.*, the first coordinate), Eq. (7) gives the following output changes:

$$\begin{aligned} \boldsymbol{\delta}_1^{(r+2)} &:= (\mathcal{F}_{\theta}(\mathbf{x} + \boldsymbol{\delta}^{(1)}) - \mathcal{F}_{\theta}(\mathbf{x}))_1 = \mathbf{G}_1^{(k+1)} \boldsymbol{\delta}^{(k+1)}, \\ \boldsymbol{\delta}'_1^{(r+2)} &:= (\mathcal{F}_{\theta}(\mathbf{x}) - \mathcal{F}_{\theta}(\mathbf{x} - \boldsymbol{\delta}^{(1)}))_1 = \mathbf{G}_1^{(k+1)} \boldsymbol{\delta}'^{(k+1)}. \end{aligned} \quad (14)$$

They satisfy $\boldsymbol{\delta}_1^{(r+2)} = \boldsymbol{\delta}'_1^{(r+2)} + \mathbf{G}_{1,j}^{(k+1)} \mathbf{A}_j^{(k)} \boldsymbol{\delta}^{(k)}$. Thus, the active side contains the additional contribution of the target neuron.

Since $|\mathbf{A}_j^{(k)} \boldsymbol{\delta}^{(k)}| = \|\mathbf{A}_j^{(k)}\| \|\boldsymbol{\delta}^{(k)}\| |\cos \angle(\mathbf{A}_j^{(k)}, \boldsymbol{\delta}^{(k)})|$, $|\mathbf{A}_j^{(k)} \boldsymbol{\delta}^{(k)}|$ is maximized when $\boldsymbol{\delta}^{(k)}$ is parallel to $\mathbf{A}_j^{(k)}$. Since $\boldsymbol{\delta}^{(k)} \in \mathbb{V}_{\mathbf{x}}^{(k)}$ according to Def. 6 and $\mathbf{A}_j^{(k)} \notin$

$\mathbb{V}_{\mathbf{x}}^{(k)}$, the *Neuron Wiggle* thereby chooses the direction of the wiggle $\delta^{(1)} \in \mathbb{R}^{d^{(1)}}$, to have a layer- k wiggle $\delta^{(k)}$, which is exactly parallel to the projection of $\mathbf{A}_j^{(k)}$ to $\mathbb{V}_{\mathbf{x}}^{(k)}$. This wiggle strengthens the target neuron (k, j) 's contribution, making the absolute network output change on the active side more likely to be larger. It therefore predicts the side with the larger absolute output change to be the target active side [4]. Since a single comparison depends on several factors, notably the network architecture and the neuron activation states around $\mathbf{x}$, an individual vote is not guaranteed to be correct. *Neuron Wiggle* thereby aggregates votes from many critical points to improve the accuracy of sign recovery.

Motivation from *Neuron Wiggle: Future Toggle* [5]. In the hard-label setting, the attacker cannot observe changes in the output logits. At EURO-CRYPT 2025, Carlini *et al.* [5] instead compare the walking distance from the two sides of a dual point to the first future-layer neuron toggle. When a boundary walk crosses a neuron's activation boundary, its activation state changes and the visible decision boundary bends, as illustrated in Fig. 2. We call such an activation-state change a *neuron toggle*. The side that reaches a future-layer toggle after a shorter distance is predicted as the target active side.

The distance comparison is motivated by the statistical signal exploited by *Neuron Wiggle*. Conceptually, consider two opposite input perturbations $\delta^{(1)}$ and $\delta'^{(1)} = -\delta^{(1)}$ from the dual point $\mathbf{x}$. The corresponding layer- $(k+1)$ wiggles $\delta^{(k+1)}$ and $\delta'^{(k+1)}$ satisfy Eq. (12). For a future neuron (p, q) with $p > k$, the corresponding changes in its preactivation on the two sides are $\left| \mathbf{A}_q^{(p)} \mathbf{I}^{(p-1)} \dots \mathbf{A}^{(k+1)} \delta^{(k+1)} \right|$ and $\left| \mathbf{A}_q^{(p)} \mathbf{I}^{(p-1)} \dots \mathbf{A}^{(k+1)} \delta'^{(k+1)} \right|$ respectively. For the same distance $\|\delta^{(1)}\|$, the $\|\delta^{(k+1)}\|$ on the active side is larger than $\|\delta'^{(k+1)}\|$ on the inactive side according to Eq. (13), hence a future neuron's preactivation tends to change faster on the active side. Although hard-label access doesn't reveal preactivation's rate of change, it reveals the resulting activation-state change when the preactivation crosses zero. *Future Toggle* therefore uses the distance to this toggle as an indirect proxy for the unobservable rate of change.

To strengthen this effect, the input perturbation should maximize $\left| \mathbf{A}_j^{(k)} \mathbf{F}^{(k-1)} \delta^{(1)} \right|$. The ideal input-space direction is therefore parallel to $(\mathbf{A}_j^{(k)} \mathbf{F}^{(k-1)})^\top$. However, the attacker must remain on the visible decision boundary in order to detect its bends. For an adjacent decision facet with unit normal $\mathbf{n}^{(1)}$, the walking direction is thus chosen parallel to $(\mathbf{A}_j^{(k)} \mathbf{F}^{(k-1)})^\top - \left\langle (\mathbf{A}_j^{(k)} \mathbf{F}^{(k-1)})^\top, \mathbf{n}^{(1)} \right\rangle \mathbf{n}^{(1)}$.

Starting from the dual point $\mathbf{x}$, the attacker follows the projected direction on one adjacent decision facet until the decision boundary bends. Using the recovered parameters of layers $1, \dots, k$, the attacker checks whether the bend is caused by a neuron in one of these layers. If so, it adds the current segment length (i.e., the Euclidean distance from the previous bend, or from $\mathbf{x}$ for the first segment) to the accumulated distance, relocates onto the adjacent decision facet, recomputes the projected direction, and continues walking. Otherwise, the bend is attributed to a future-layer neuron toggle and the walk terminates. The

same procedure is applied on the other side of $\mathbf{x}$, and the side with the shorter accumulated distance is predicted as the target active side.

Limitations of the *Future Toggle*. *Future Toggle* [5] attempts to identify the target active side by comparing the distances from a dual point to the first future layer toggles on its two sides. However, obtaining these distances and using them for sign recovery introduce limitations in both efficiency and accuracy.

To facilitate our discussion, we perform a sign recovery experiment using *Future Toggle* method [5] on the CIFAR-10 network with architecture 3072-256×3-64-10. The white-box information is used to identify bends caused by neurons in the recovered layers and to evaluate whether each valid vote is correct. Table 2 summarizes the resulting boundary walking and voting statistics. The column of “Bends from layers $1, \dots, k$ ” reports the number of bends (neuron toggles) from recovered layers encountered by walking from each attempted dual point. Entries in Table 2 are reported as “the mean $\pm$ standard deviation” values across the ten selected neurons.

Table 2: White-box diagnostic of *Future Toggle* on the CIFAR-10 network 3072-256×3-64-10. For each hidden layer, ten neurons are randomly selected, with 1000 attempted dual points evaluated for each of these neurons.

Layer	Bends from layers $1, \dots, k$	Dual point utilization (%) $\eta_{\text{dual}} = n_{\text{valid}}/n_{\text{attempt}}$	Vote accuracy (%) $p = n_{\text{correct}}/n_{\text{valid}}$
1	0.87 ± 0.22	99.74 ± 0.18	54.14 ± 7.16
2	1.58 ± 0.08	94.65 ± 1.63	56.51 ± 3.15
3	6.73 ± 0.25	68.18 ± 2.63	55.31 ± 2.31
4	12.45 ± 0.52	3.21 ± 0.53	52.82 ± 8.08

- **Limitation 1: Cost of decision boundary tracing.** During each of the two walks in $\mathbb{D}_{ab} \cap \mathbb{L}_{\mathbf{x}}$ and $\mathbb{D}_{ab} \cap \mathbb{L}'_{\mathbf{x}}$ from a dual point $\mathbf{x}$, the attacker must detect when the current decision facet ends and the decision boundary bends. This information is not directly provided by the hard-label oracle. Instead, the attacker must repeatedly query the oracle to determine whether the walk remains on the same decision facet and use binary search to locate the bend when the facet changes. This cost is further amplified when a detected bend is caused by the neuron in a known layer $1, \dots, k$ rather than by a future layer. Such a bend does not terminate the walk. Instead, the attacker must recover the normal of the new decision facet and continue the walk. Each such bend requires the recovery of a new decision facet normal, which involves $d^{(1)} - 1$ coordinate ratio searches, according to Eq. (22) in Sect. 4.1. Consequently, this cost can be substantial when the model input dimension is large (e.g., $d^{(1)} = 3072$).

In the 2nd column of Table 2, when the layer depth increases, the mean number of bends from the recovered layers rises from 0.87 to 12.45, indicating that walks in deeper layers require more decision-facet normal recoveries.

- **Limitation 2: Limited utilization of dual points.** A search may repeatedly encounter activation boundaries belonging to recovered layers or fail to reach a future layer toggle within the maximum searching distance. In either case, the dual point is discarded. Let n_{attempt} denote the number of attempted dual points and n_{valid} the number that produce valid votes. The dual point utilization rate is defined as $\eta_{\text{dual}} := n_{\text{valid}}/n_{\text{attempt}}$. A lower η_{dual} requires more attempted dual points and results in a longer time. In the third column of Table 2, as the layer depth increases, the dual point utilization rate falls from 99.74% to 3.21%.
- **Limitation 3: Weak statistical advantage.** *Future Toggle* infers the target active side through two successive proxy relations: a larger target layer output change is expected to produce faster changes in future neurons, and the faster neuron value changes are expected to produce a shorter distance to the first future layer toggle. Neither relation is guaranteed to hold at every dual point, and the probability that the target active side produces the shorter distance may be only slightly greater than one half. Let $p = 1/2 + \gamma$ denote the probability that an individual valid vote is correct, where $\gamma > 0$ represents the statistical advantage in favor of the target active side. Assuming that the valid votes are independent and share the same success probability p , Hoeffding’s inequality gives $\Pr[\text{the majority vote is incorrect}] \leq \exp(-2\gamma^2 n_{\text{valid}})$. Therefore, ensuring an error probability of at most α requires $n_{\text{valid}} \geq \ln(1/\alpha)/(2\gamma^2)$, which grows with $1/\gamma^2$. For example, when $p = 0.55$, about $n_{\text{valid}} = 10^3$ valid votes are required to reach a confidence level of 99% [5]. This bound shows that a weaker statistical advantage requires more valid votes to reach the same confidence level. With a limited budget n_{attempt} , more neurons may therefore remain below the required confidence threshold. In the 4th column of Table 2, n_{correct} denotes the number of valid dual points that produce correct votes; the vote accuracy is $p = n_{\text{correct}}/n_{\text{valid}}$. The average vote accuracy of each layer ranges from 52.82% to 56.51%. It shows that an individual valid vote is only slightly more likely to be correct than random guessing.
- **Limitation 4: Amplified limitations in the last hidden layer.** The *Future Toggle* requires a different terminal event for the last hidden layer, where there is no future-layer ReLU neuron. It therefore continues each search until an intersection of decision boundaries and uses this distance instead [5]. This special treatment amplifies the preceding three limitations. First, a class decision boundary intersection may be farther from the dual point than a future layer neuron toggle, resulting in a longer walk. Second, the required intersection may not be reached within the maximum walking distance and the corresponding dual point is then discarded, reducing η_{dual} and increasing the number n_{attempt} of attempted dual points. Finally, the two walks may terminate at intersections with different class decision boundaries, so their

measured distances depend on different class decision boundaries as well as on the rates of logit change.

In the last hidden layer of Table 2, only about 32 of the 1000 attempted dual points per neuron produce valid votes on average.

3.2 System of Equations (SOE) and Its Extensions

The Raw-Output SOE Method [4]. At EUROCRYPT 2024, Canales-Martínez *et al.* proposed the System of Equations (SOE) method in S5 setting. Specifically, targeting a single coordinate of the output vector (*e.g.*, the first coordinate) at an input $\mathbf{x}$, the attacker samples $d^{(k+1)}$ perturbations $\boldsymbol{\delta}_1^{(1)}, \dots, \boldsymbol{\delta}_{d^{(k+1)}}^{(1)} \in \mathbb{R}^{d^{(1)}}$ such that every perturbed input $\mathbf{x} + \boldsymbol{\delta}_\ell^{(1)}$ ($\ell \in [d^{(k+1)}]$) remains within the same linear neighborhood as $\mathbf{x}$. Then the linear system is built by computing the output difference between $\mathcal{F}_\theta(\mathbf{x} + \boldsymbol{\delta}_\ell)$ and $\mathcal{F}_\theta(\mathbf{x})$,

$$\begin{bmatrix} \left(\hat{\mathbf{A}}^{(k)} \mathbf{F}^{(k-1)} \boldsymbol{\delta}_1^{(1)} \right)^\top \\ \vdots \\ \left(\hat{\mathbf{A}}^{(k)} \mathbf{F}^{(k-1)} \boldsymbol{\delta}_{d^{(k+1)}}^{(1)} \right)^\top \end{bmatrix} \left(\mathbf{G}_1^{(k+1)} \mathbf{I}^{(k)} \mathbf{S}^{(k)} \right)^\top = \begin{bmatrix} (\mathcal{F}_\theta(\mathbf{x} + \boldsymbol{\delta}_1^{(1)}) - \mathcal{F}_\theta(\mathbf{x}))_1 \\ \vdots \\ (\mathcal{F}_\theta(\mathbf{x} + \boldsymbol{\delta}_{d^{(k+1)}}^{(1)}) - \mathcal{F}_\theta(\mathbf{x}))_1 \end{bmatrix}, \quad (15)$$

where $\mathbf{F}^{(k-1)}$, $\hat{\mathbf{A}}^{(k)}$, $\hat{\mathbf{b}}^{(k)}$ are known and the target is to recover $\mathbf{S}^{(k)}$. Solving this linear system yields the unknown vector $(\mathbf{G}_1^{(k+1)} \mathbf{I}^{(k)} \mathbf{S}^{(k)})^\top$. Since the ReLU activation suppresses negative values, any neuron t in the layer k that is inactive at $\mathbf{x}$ will have a corresponding entry of zero in $(\mathbf{G}_1^{(k+1)} \mathbf{I}^{(k)} \mathbf{S}^{(k)})^\top$. Once the inactive neurons are identified (*e.g.*, neuron t), if $\hat{\mathbf{A}}_t^{(k)} \mathbf{x}^{(k)} + \hat{\mathbf{b}}_t^{(k)} > 0$, its sign will be $s_t^{(k)} = -1$. To ensure the linear system in Eq. (15) has a unique solution, the coefficient matrix must be of full rank, which requires $\text{rank}(\hat{\mathbf{A}}^{(k)} \mathbf{F}^{(k-1)}) = d^{(k+1)}$. This implies $d^{(1)}, \dots, d^{(k)} \geq d^{(k+1)}$. Consequently, this deterministic method is primarily applicable to network architectures that are sufficiently contractive.

The SOE + Wiggle in Raw-Output Setting [18]. At EUROCRYPT 2026, Liu *et al.* combined the SOE and the *Neuron Wiggle* methods. As summarized in Sect. 3.1, *Neuron Wiggle* [4] recovers the sign of each neuron along with a confidence level, where a high confidence level strongly indicates a correct recovery. Recall that the unknown vector in Eq. (15) is expressed as $(\mathbf{G}_1^{(k+1)} \mathbf{I}^{(k)} \mathbf{S}^{(k)})^\top = (\mathbf{G}_{1,1}^{(k+1)} \tau_1^{(k)} s_1^{(k)}, \dots, \mathbf{G}_{1,d^{(k+1)}}^{(k+1)} \tau_{d^{(k+1)}}^{(k)} s_{d^{(k+1)}}^{(k)})^\top$. If neuron t is identified as inactive (determined by evaluating $s_t^{(k)}(\hat{\mathbf{A}}_t^{(k)} \mathbf{x}^{(k)} + \hat{\mathbf{b}}_t^{(k)}) < 0$, provided that the sign $s_t^{(k)}$ is recovered correctly with a high confidence level by *Neuron Wiggle*), the corresponding term $\mathbf{G}_{1,t}^{(k+1)} \tau_t^{(k)} s_t^{(k)}$ can be directly set to 0, thereby reducing the number of unknowns. Let $\mathbb{K}$ contain the neurons that *Neuron Wiggle* identifies as inactive at $\mathbf{x}$ with high confidence, and $\mathbb{U} := [d^{(k+1)}] \setminus \mathbb{K}$ contain the remaining neurons. For a vector $\mathbf{v}$, let $[\mathbf{v}]_{\mathbb{K}}$ and $[\mathbf{v}]_{\mathbb{U}}$ denote the subvectors indexed by

$\mathbb{K}$ and $\mathbb{U}$, respectively. Since every neuron in $\mathbb{K}$ is inactive, the corresponding entries of $(\mathbf{G}_1^{(k+1)} \mathbf{I}^{(k)} \mathbf{S}^{(k)})^\top$ are zero. The *SOE* thereby reduces to

$$\begin{bmatrix} \left[\hat{\mathbf{A}}^{(k)} \mathbf{F}^{(k-1)} \boldsymbol{\delta}_1^{(1)} \right]_{\mathbb{U}}^\top \\ \vdots \\ \left[\hat{\mathbf{A}}^{(k)} \mathbf{F}^{(k-1)} \boldsymbol{\delta}_{d^{(k+1)}}^{(1)} \right]_{\mathbb{U}}^\top \end{bmatrix} \left[\left(\mathbf{G}_1^{(k+1)} \mathbf{I}^{(k)} \mathbf{S}^{(k)} \right)^\top \right]_{\mathbb{U}} = \begin{bmatrix} (\mathcal{F}_\theta(\mathbf{x} + \boldsymbol{\delta}_1^{(1)}) - \mathcal{F}_\theta(\mathbf{x}))_1 \\ \vdots \\ (\mathcal{F}_\theta(\mathbf{x} + \boldsymbol{\delta}_{d^{(k+1)}}^{(1)}) - \mathcal{F}_\theta(\mathbf{x}))_1 \end{bmatrix}. \quad (16)$$

The total number of unknowns reduces from $d^{(k+1)}$ to $|\mathbb{U}|$. For these remaining unknowns to be uniquely determined, the rank of the coefficient matrix in Eq. (16) must equal $|\mathbb{U}|$. Since the rank of this coefficient matrix is upper-bounded by $\text{rank}(\hat{\mathbf{A}}^{(k)} \mathbf{F}^{(k-1)})$, then $|\mathbb{U}| \leq \text{rank}(\hat{\mathbf{A}}^{(k)} \mathbf{F}^{(k-1)})$. Since $|\mathbb{U}| + |\mathbb{K}| = d^{(k+1)}$, this yields the following necessary condition for uniquely solving Eq. (16):

$$|\mathbb{K}| \geq d^{(k+1)} - \text{rank}(\hat{\mathbf{A}}^{(k)} \mathbf{F}^{(k-1)}). \quad (17)$$

The original *SOE* method [4] constructs the linear system at a single input $\mathbf{x}$, which inherently limits the rank of the system to the rank of the local prefix matrix $\mathbf{F}^{(k-1)}$ around $\mathbf{x}$. However, since the unknown vector is $(\mathbf{G}_1^{(k+1)} \mathbf{I}^{(k)} \mathbf{S}^{(k)})^\top$, any input point sharing the same activation state in the layers k to $r+1$ can be utilized. In other words, as long as the activation states of all neurons in layer k and all subsequent layers remain invariant, the target layer's activation matrix $\mathbf{I}^{(k)}$ and the suffix map $\mathbf{G}^{(k+1)}$ are identical across these inputs. Meanwhile, the activation states of neurons in layers $1, \dots, k-1$ may differ across these inputs. These differences can produce distinct prefix maps $\mathbf{F}_{\mathbf{x}_t}^{(k-1)}$ for $\mathbf{x}_t$ ($1 \leq t \leq T$), and hence distinct *SOE* coefficient matrices. At each $\mathbf{x}_t$ ($1 \leq t \leq T$), choose perturbations $\boldsymbol{\delta}_{t,1}^{(1)}, \dots, \boldsymbol{\delta}_{t,d^{(k+1)}}^{(1)}$ that remain in its linear neighborhood. After removing the entries indexed by $\mathbb{K}$, the corresponding *SOE* systems can be combined as

$$\begin{bmatrix} \left[\hat{\mathbf{A}}^{(k)} \mathbf{F}_{\mathbf{x}_1}^{(k-1)} \boldsymbol{\delta}_{1,1}^{(1)} \right]_{\mathbb{U}}^\top \\ \vdots \\ \left[\hat{\mathbf{A}}^{(k)} \mathbf{F}_{\mathbf{x}_1}^{(k-1)} \boldsymbol{\delta}_{1,d^{(k+1)}}^{(1)} \right]_{\mathbb{U}}^\top \\ \vdots \\ \left[\hat{\mathbf{A}}^{(k)} \mathbf{F}_{\mathbf{x}_T}^{(k-1)} \boldsymbol{\delta}_{T,1}^{(1)} \right]_{\mathbb{U}}^\top \\ \vdots \\ \left[\hat{\mathbf{A}}^{(k)} \mathbf{F}_{\mathbf{x}_T}^{(k-1)} \boldsymbol{\delta}_{T,d^{(k+1)}}^{(1)} \right]_{\mathbb{U}}^\top \end{bmatrix} \left[\left(\mathbf{G}_1^{(k+1)} \mathbf{I}^{(k)} \mathbf{S}^{(k)} \right)^\top \right]_{\mathbb{U}} = \begin{bmatrix} (\mathcal{F}_\theta(\mathbf{x}_1 + \boldsymbol{\delta}_{1,1}^{(1)}) - \mathcal{F}_\theta(\mathbf{x}_1))_1 \\ \vdots \\ (\mathcal{F}_\theta(\mathbf{x}_1 + \boldsymbol{\delta}_{1,d^{(k+1)}}^{(1)}) - \mathcal{F}_\theta(\mathbf{x}_1))_1 \\ \vdots \\ (\mathcal{F}_\theta(\mathbf{x}_T + \boldsymbol{\delta}_{T,1}^{(1)}) - \mathcal{F}_\theta(\mathbf{x}_T))_1 \\ \vdots \\ (\mathcal{F}_\theta(\mathbf{x}_T + \boldsymbol{\delta}_{T,d^{(k+1)}}^{(1)}) - \mathcal{F}_\theta(\mathbf{x}_T))_1 \end{bmatrix}. \quad (18)$$

The combined coefficient matrix may have a higher rank than the coefficient matrix obtained at any individual input. When its rank reaches $|\mathbb{U}|$, the remaining unknown entries are uniquely determined.

Hard-Label SOE [3]. Canales-Martínez *et al.* extended the *SOE* method [4] to S1 setting at LATINCRYPT 2025. Suppose a transition point $\mathbf{x} \in \mathbb{D}_{ab}$ is located at the decision boundary between classes a and b ($a < b$). According to Eq. (8) and (6), we have:

$$\mathcal{D}_{ab}(\mathbf{x}) = \left(\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)} \right) \mathbf{x}^{(k+1)} + \gamma_a^{(k+1)} - \gamma_b^{(k+1)} = 0, \quad (19)$$

where $\mathbf{x}^{(k+1)} = \mathbf{I}^{(k)}(\mathbf{A}^{(k)}(\mathbf{F}^{(k-1)}\mathbf{x} + \boldsymbol{\beta}^{(k-1)}) + \mathbf{b}^{(k)})$. The attacker then samples $d^{(k+1)} - 1$ perturbations $\boldsymbol{\delta}_1^{(1)}, \dots, \boldsymbol{\delta}_{d^{(k+1)}-1}^{(1)}$ such that $\mathbf{x} + \boldsymbol{\delta}_\ell^{(1)} \in \mathbb{D}_{ab} \cap \mathbb{L}_{\mathbf{x}}$, with $\ell \in [d^{(k+1)} - 1]$. Then according to Eq. (19), we have

$$\mathcal{D}_{ab}(\mathbf{x} + \boldsymbol{\delta}_\ell^{(1)}) - \mathcal{D}_{ab}(\mathbf{x}) = \left(\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)} \right) \mathbf{I}^{(k)} \mathbf{S}^{(k)} \hat{\mathbf{A}}^{(k)} \mathbf{F}^{(k-1)} \boldsymbol{\delta}_\ell^{(1)} = 0. \quad (20)$$

With $d^{(k+1)} - 1$ linearly independent equations from $\boldsymbol{\delta}_1^{(1)}, \dots, \boldsymbol{\delta}_{d^{(k+1)}-1}^{(1)}$, we can construct the linear system,

$$\begin{bmatrix} \left[\hat{\mathbf{A}}^{(k)} \mathbf{F}^{(k-1)} \boldsymbol{\delta}_1^{(1)} \right]^\top \\ \vdots \\ \left[\hat{\mathbf{A}}^{(k)} \mathbf{F}^{(k-1)} \boldsymbol{\delta}_{d^{(k+1)}-1}^{(1)} \right]^\top \end{bmatrix} \left[\left((\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)}) \mathbf{I}^{(k)} \mathbf{S}^{(k)} \right)^\top \right] = \mathbf{0}. \quad (21)$$

Solving Eq. (21) yields the unknown vector $\left((\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)}) \mathbf{I}^{(k)} \mathbf{S}^{(k)} \right)^\top$. Then the sign recovery process is similar to *SOE* method introduced above.

Limitation. To ensure the linear system in Eq. (21) has a unique nonzero solution up to a scalar multiple, also the same as the *SOE* method, hard label *SOE* requires $\text{rank}(\hat{\mathbf{A}}^{(k)} \mathbf{F}^{(k-1)}) = d^{(k+1)}$, then $d^{(1)}, \dots, d^{(k)} \geq d^{(k+1)}$. Consequently, the network architectures also need to be *strongly contractive*.

4 Hard-Label Sign Recovery via *Normal Alignment*

4.1 Recovering Decision-Facet Normals in S5/S1 Settings

Recovering Unit Decision-Facet Normals in the Model Input Space Using [7,5]. Given a transition point $\mathbf{x} \in \mathbb{D}_{ab}$, recall from Eq. (19), we have $\mathcal{D}_{ab}(\mathbf{x}) = \left(\mathbf{G}_a^{(1)} - \mathbf{G}_b^{(1)} \right) \mathbf{x}^{(1)} + \gamma_a^{(1)} - \gamma_b^{(1)} = 0$. The normal vector $\mathbf{g}^{(1)}$ of the local decision facet $\mathbb{D}_{ab} \cap \mathbb{L}_{\mathbf{x}}$, is given by $\mathbf{g}^{(1)} := (\mathbf{G}_a^{(1)} - \mathbf{G}_b^{(1)})^\top \in \mathbb{R}^{d^{(1)}}$.

The $\mathbf{g}^{(1)}$ can be recovered up to a nonzero scalar using hard-label queries following the methods in [7,5]. Let $\mathbf{e}_1, \dots, \mathbf{e}_{d^{(1)}}$ denote the standard basis of the model input space $\mathbb{R}^{d^{(1)}}$. For each $\ell \in [d^{(1)}]$, let $\mathbf{g}_\ell^{(1)} := \langle \mathbf{g}^{(1)}, \mathbf{e}_\ell \rangle$ denote the ℓ -th coordinate of $\mathbf{g}^{(1)}$. Suppose that $\mathbf{g}_1^{(1)} \neq 0$. For each $\ell \in [d^{(1)}]$, take a sufficiently small step $\alpha \mathbf{e}_\ell$ and use hard-label binary search along $\mathbf{e}_1$ to find

a scalar β_ℓ such that $\mathbf{x} + \alpha \mathbf{e}_\ell + \beta_\ell \mathbf{e}_1$ returns to the decision boundary $\mathbb{D}_{ab}$. Provided that the returned point remains on the same affine decision facet, then we have $\alpha \mathbf{g}_\ell^{(1)} + \beta_\ell \mathbf{g}_1^{(1)} = 0$, and hence $\mathbf{g}_\ell^{(1)} / \mathbf{g}_1^{(1)} = -\beta_\ell / \alpha$. Using these recovered coordinate ratios, we can recover:

$$\hat{\mathbf{g}}^{(1)} := \left(1, -\frac{\beta_2}{\alpha}, \dots, -\frac{\beta_{d(1)}}{\alpha}\right) = \frac{\mathbf{g}^{(1)}}{\mathbf{g}_1^{(1)}}. \quad (22)$$

Although its magnitude cannot be recovered, its sign can be determined with the following method. For a sufficiently small $\varepsilon > 0$, $\mathbf{x} + \varepsilon \hat{\mathbf{g}}^{(1)}$ is still in the linear neighborhood $\mathbb{L}_\mathbf{x}$, we have

$$\mathcal{D}_{ab}(\mathbf{x} + \varepsilon \hat{\mathbf{g}}^{(1)}) - \mathcal{D}_{ab}(\mathbf{x}) = \mathbf{g}^{(1)} \cdot \varepsilon \hat{\mathbf{g}}^{(1)} = \varepsilon \langle \mathbf{g}^{(1)}, \hat{\mathbf{g}}^{(1)} \rangle. \quad (23)$$

- If the hard-label oracle returns class a at $\mathbf{x} + \varepsilon \hat{\mathbf{g}}^{(1)}$, then $\mathcal{D}_{ab}(\mathbf{x} + \varepsilon \hat{\mathbf{g}}^{(1)}) - \mathcal{D}_{ab}(\mathbf{x}) > 0$, i.e., $\langle \mathbf{g}^{(1)}, \hat{\mathbf{g}}^{(1)} \rangle > 0$. The sign of $\hat{\mathbf{g}}^{(1)}$ follows $\mathbf{g}^{(1)}$.
- In contrast, if it returns class b , then $\langle \mathbf{g}^{(1)}, \hat{\mathbf{g}}^{(1)} \rangle < 0$, and $\hat{\mathbf{g}}^{(1)}$ needs to be reversed.

After normalization, we obtain the *unit decision-facet normal* of $\mathbb{D}_{ab} \cap \mathbb{L}_\mathbf{x}$ as $\mathbf{n}^{(1)} := \mathbf{g}^{(1)} / \|\mathbf{g}^{(1)}\|$.

Projected Decision-Facet Normal in the Input Space of Layer k . Let $\mathbf{g}^{(k)}$ denote the *decision-facet normal in the input space of layer $k \geq 1$* , then $\mathbf{g}^{(k)} = \left[\left(\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)} \right) \mathbf{I}^{(k)} \mathbf{A}^{(k)} \right]^\top \in \mathbb{R}^{d^{(k)}}$. Note that the decision-facet normal $\mathbf{g}^{(1)}$ in the model input space and the decision-facet normal $\mathbf{g}^{(k)}$ in the input space of layer k satisfy:

$$\mathbf{g}^{(1)} = (\mathbf{F}^{(k-1)})^\top \mathbf{g}^{(k)}. \quad (24)$$

This constructs a linear system with an unknown vector $\mathbf{g}^{(k)}$. Then we can recover $\mathbf{g}^{(k)}$ from $\mathbf{g}^{(1)}$ by solving this system. If $\text{rank}(\mathbf{F}^{(k-1)}) = d^{(k)}$, then the system has the unique solution $\mathbf{g}^{(k)}$. When $\text{rank}(\mathbf{F}^{(k-1)}) < d^{(k)}$, the system admits multiple solutions, and its solution set is $\mathbf{g}^{(k)} + \ker((\mathbf{F}^{(k-1)})^\top)$. Although the full $\mathbf{g}^{(k)}$ is not uniquely determined in the latter case, its projection $\bar{\mathbf{g}}^{(k)} = \mathbf{P}^{(k)} \mathbf{g}^{(k)}$ can be recovered as the unique minimum-norm solution of the system using least squares as proved below.

Let $\mathbf{h}$ be any solution to Eq. (24), so that $\mathbf{h} = \mathbf{g}^{(k)} + \mathbf{u}$ for some $\mathbf{u} \in \ker((\mathbf{F}^{(k-1)})^\top)$. By Def. 7, $\ker((\mathbf{F}^{(k-1)})^\top) = (\mathbb{V}_\mathbf{x}^{(k)})^\perp$. Since $\mathbf{P}^{(k)}$ is the orthogonal projector onto $\mathbb{V}_\mathbf{x}^{(k)}$, we have $\mathbf{g}^{(k)} - \mathbf{P}^{(k)} \mathbf{g}^{(k)} \in \ker((\mathbf{F}^{(k-1)})^\top)$. Therefore, we can rewrite $\mathbf{h}$ as

$$\mathbf{h} = \mathbf{P}^{(k)} \mathbf{g}^{(k)} + \left(\mathbf{g}^{(k)} - \mathbf{P}^{(k)} \mathbf{g}^{(k)} + \mathbf{u} \right) = \mathbf{P}^{(k)} \mathbf{g}^{(k)} + \mathbf{u}', \quad (25)$$

where $\mathbf{u}' = \mathbf{g}^{(k)} - \mathbf{P}^{(k)} \mathbf{g}^{(k)} + \mathbf{u} \in \ker((\mathbf{F}^{(k-1)})^\top)$. Since $\mathbf{P}^{(k)} \mathbf{g}^{(k)} \in \mathbb{V}_\mathbf{x}^{(k)}$ and $\mathbf{u}' \in (\mathbb{V}_\mathbf{x}^{(k)})^\perp$, the two vectors are orthogonal. Consequently, $\|\mathbf{h}\|^2 = \|\mathbf{P}^{(k)} \mathbf{g}^{(k)}\|^2 +$

$\|\mathbf{u}'\|^2$. The norm is therefore uniquely minimized when $\mathbf{u}' = \mathbf{0}$. Thus, the projected decision-facet normal $\bar{\mathbf{g}}^{(k)} = \mathbf{P}^{(k)} \mathbf{g}^{(k)}$ is the unique minimum-norm solution of Eq. (24).

In the hard-label setting, the attacker recovers the *unit decision-facet normal* $\mathbf{n}^{(1)} = \mathbf{g}^{(1)} / \|\mathbf{g}^{(1)}\|$ rather than $\mathbf{g}^{(1)}$. We therefore solve $(\mathbf{F}^{(k-1)})^\top \frac{\mathbf{g}^{(k)}}{\|\mathbf{g}^{(1)}\|} = \mathbf{n}^{(1)}$ and choose any solution $\frac{\mathbf{g}^{(k)}}{\|\mathbf{g}^{(1)}\|} \in \mathbb{R}^{d^{(k)}}$. Following the same argument as above, all such solutions have the same projection onto the control space. Given any solution $\mathbf{g}^{(k)} / \|\mathbf{g}^{(1)}\|$, this common projection is $\mathbf{P}^{(k)} \frac{\mathbf{g}^{(k)}}{\|\mathbf{g}^{(1)}\|} = \frac{\bar{\mathbf{g}}^{(k)}}{\|\mathbf{g}^{(1)}\|}$. Normalizing this projection gives

$$\bar{\mathbf{n}}^{(k)} := \frac{\bar{\mathbf{g}}^{(k)}}{\|\bar{\mathbf{g}}^{(k)}\|}. \quad (26)$$

4.2 Normal Lengths Comparison in Raw-Output Setting

Two-Side Projected Decision-Facet Normals. Let $\mathbf{x} \in \mathbb{C}_j^{(k)} \cap \mathbb{D}_{ab}$ be a dual point, and let $\mathbb{L}_{\mathbf{x}}$ and $\mathbb{L}'_{\mathbf{x}}$ denote the two linear neighborhoods adjacent to the target critical hyperplane. Then the neuron activation states in $\mathbb{L}_{\mathbf{x}}$ and $\mathbb{L}'_{\mathbf{x}}$ differ only in the state of neuron (k, j) . Without loss of generality, suppose that the target neuron is active in $\mathbb{L}_{\mathbf{x}}$ and inactive in $\mathbb{L}'_{\mathbf{x}}$. Denote the corresponding activation matrices by $\mathbf{I}_+^{(k)}$ and $\mathbf{I}_-^{(k)}$, and the decision-facet normals in the input space of layer k by $\mathbf{g}^{(k)}$ and $\mathbf{g}'^{(k)}$, respectively. Then we have,

$$\begin{aligned} \mathbf{g}^{(k)} &:= \left[\left(\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)} \right) \mathbf{I}_+^{(k)} \mathbf{A}^{(k)} \right]^\top \\ &= \sum_{i \in \mathbb{S} \setminus \{j\}} (\mathbf{G}_{a,i}^{(k+1)} - \mathbf{G}_{b,i}^{(k+1)}) (\mathbf{A}_i^{(k)})^\top + (\mathbf{G}_{a,j}^{(k+1)} - \mathbf{G}_{b,j}^{(k+1)}) (\mathbf{A}_j^{(k)})^\top, \\ \mathbf{g}'^{(k)} &:= \left[\left(\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)} \right) \mathbf{I}_-^{(k)} \mathbf{A}^{(k)} \right]^\top = \sum_{i \in \mathbb{S} \setminus \{j\}} (\mathbf{G}_{a,i}^{(k+1)} - \mathbf{G}_{b,i}^{(k+1)}) (\mathbf{A}_i^{(k)})^\top, \end{aligned} \quad (27)$$

where $\mathbb{S}$ contains the indices of all active neurons in layer k . Therefore, the difference between $\mathbf{g}^{(k)}$ and $\mathbf{g}'^{(k)}$ is exactly a scalar multiple of the target neuron's weight $\mathbf{A}_j^{(k)}$, *i.e.*,

$$\mathbf{g}^{(k)} - \mathbf{g}'^{(k)} = (\mathbf{G}_{a,j}^{(k+1)} - \mathbf{G}_{b,j}^{(k+1)}) (\mathbf{A}_j^{(k)})^\top. \quad (28)$$

According to Eq. (24) in Sect. 4.1, $\mathbf{g}^{(k)}$ (or $\mathbf{g}'^{(k)}$) can be recovered from the system $\mathbf{g}^{(1)} = (\mathbf{F}^{(k-1)})^\top \mathbf{g}^{(k)}$. In the deeper hidden layers, $\mathbf{F}^{(k-1)}$ is often rank deficient, so the full $\mathbf{g}^{(k)}$ (or $\mathbf{g}'^{(k)}$) is generally not uniquely recoverable. Only their projections onto the control space $\mathbb{V}_{\mathbf{x}}^{(k)}$, $\bar{\mathbf{g}}^{(k)} := \mathbf{P}^{(k)} \mathbf{g}^{(k)}$ and $\bar{\mathbf{g}}'^{(k)} := \mathbf{P}^{(k)} \mathbf{g}'^{(k)}$, are uniquely determined.

By Def. 7, $\mathbf{P}^{(k)}$ is symmetric. Hence, the projected decision-facet normals can be written as

$$\begin{aligned}
\bar{\mathbf{g}}^{(k)} &:= \mathbf{P}^{(k)} \mathbf{g}^{(k)} = \left[\left(\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)} \right) \mathbf{I}_+^{(k)} \mathbf{A}^{(k)} \mathbf{P}^{(k)} \right]^\top \\
&= \sum_{i \in \mathbb{S}/\{j\}} \left(\mathbf{G}_{a,i}^{(k+1)} - \mathbf{G}_{b,i}^{(k+1)} \right) \mathbf{P}^{(k)} \left(\mathbf{A}_i^{(k)} \right)^\top + \left(\mathbf{G}_{a,j}^{(k+1)} - \mathbf{G}_{b,j}^{(k+1)} \right) \mathbf{P}^{(k)} \left(\mathbf{A}_j^{(k)} \right)^\top, \\
\bar{\mathbf{g}}'^{(k)} &:= \mathbf{P}^{(k)} \mathbf{g}'^{(k)} = \left[\left(\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)} \right) \mathbf{I}_-^{(k)} \mathbf{A}^{(k)} \mathbf{P}^{(k)} \right]^\top \\
&= \sum_{i \in \mathbb{S}/\{j\}} \left(\mathbf{G}_{a,i}^{(k+1)} - \mathbf{G}_{b,i}^{(k+1)} \right) \mathbf{P}^{(k)} \left(\mathbf{A}_i^{(k)} \right)^\top.
\end{aligned} \tag{29}$$

Statistical Length Advantage. For a matrix $\mathbf{M}$, its squared Frobenius norm is defined by $\|\mathbf{M}\|_F^2 := \sum_m \|\mathbf{M}_m\|^2 = \sum_m \sum_n \mathbf{M}_{m,n}^2$. The matrices $\mathbf{I}_+^{(k)} \mathbf{A}^{(k)} \mathbf{P}^{(k)}$ and $\mathbf{I}_-^{(k)} \mathbf{A}^{(k)} \mathbf{P}^{(k)}$ have identical rows except for the row corresponding to the target neuron, which is $\mathbf{A}_j^{(k)} \mathbf{P}^{(k)}$ on the active side and zero on the inactive side. Therefore,

$$\left\| \mathbf{I}_+^{(k)} \mathbf{A}^{(k)} \mathbf{P}^{(k)} \right\|_F^2 = \left\| \mathbf{I}_-^{(k)} \mathbf{A}^{(k)} \mathbf{P}^{(k)} \right\|_F^2 + \left\| \mathbf{A}_j^{(k)} \mathbf{P}^{(k)} \right\|^2, \tag{30}$$

which shows that the active side matrix has a larger Frobenius norm, or equivalently, greater total squared row energy.

From Eq. (29), we have $\bar{\mathbf{g}}^{(k)} = \bar{\mathbf{g}}'^{(k)} + (\mathbf{G}_{a,j}^{(k+1)} - \mathbf{G}_{b,j}^{(k+1)}) \mathbf{P}^{(k)} (\mathbf{A}_j^{(k)})^\top$. Consequently,

$$\begin{aligned}
\left\| \bar{\mathbf{g}}^{(k)} \right\|^2 - \left\| \bar{\mathbf{g}}'^{(k)} \right\|^2 &= \\
2 \left(\mathbf{G}_{a,j}^{(k+1)} - \mathbf{G}_{b,j}^{(k+1)} \right) \left\langle \bar{\mathbf{g}}'^{(k)}, \mathbf{P}^{(k)} \left(\mathbf{A}_j^{(k)} \right)^\top \right\rangle &+ \left(\mathbf{G}_{a,j}^{(k+1)} - \mathbf{G}_{b,j}^{(k+1)} \right)^2 \left\| \mathbf{P}^{(k)} \left(\mathbf{A}_j^{(k)} \right)^\top \right\|^2.
\end{aligned} \tag{31}$$

Generally, the last term in Eq. (31) is positive. If the angle between $(\mathbf{G}_{a,j}^{(k+1)} - \mathbf{G}_{b,j}^{(k+1)}) \mathbf{P}^{(k)} (\mathbf{A}_j^{(k)})^\top$ and $\bar{\mathbf{g}}'^{(k)}$ is no bigger than $\pi/2$, their inner product is non-negative. The first term is therefore nonnegative, and $\|\bar{\mathbf{g}}^{(k)}\| > \|\bar{\mathbf{g}}'^{(k)}\|$.

If the angle is greater than $\pi/2$, the first term in Eq. (31) is negative, whereas the last term remains positive. The length ordering is therefore determined by their relative magnitudes. If the last term is larger than the absolute value of the first term, then $\|\bar{\mathbf{g}}^{(k)}\| > \|\bar{\mathbf{g}}'^{(k)}\|$; otherwise, $\|\bar{\mathbf{g}}^{(k)}\| \leq \|\bar{\mathbf{g}}'^{(k)}\|$.

Statistical interpretation. We formalize the preceding intuition using an idealized model.

Proposition 1. *We fix $\mathbf{I}_+^{(k)} \mathbf{A}^{(k)} \mathbf{P}^{(k)}$ and $\mathbf{I}_-^{(k)} \mathbf{A}^{(k)} \mathbf{P}^{(k)}$, and treat the suffix coefficients $\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)}$ as random. Specifically, we assume that their coordinates are independent zero-mean Gaussian variables with common variance*

$\sigma^2 > 0$ ³. We have the expectation

$$\mathbb{E} \left[\|\bar{\mathbf{g}}^{(k)}\|^2 - \|\bar{\mathbf{g}}'^{(k)}\|^2 \right] = \sigma^2 \|\mathbf{A}_j^{(k)} \mathbf{P}^{(k)}\|^2. \quad (32)$$

Proof. Let $\mathbf{g}^{(k+1)} = \mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)}$ with $\mathbf{g}_i^{(k+1)}$ denoting its i -th coordinate. For a fixed matrix $\mathbf{M}$, let $\mathbf{M}_i$ denote its i -th row. Since $\mathbf{g}^{(k+1)} \mathbf{M} = \sum_i \mathbf{g}_i^{(k+1)} \mathbf{M}_i$, bilinearity of the inner product and linearity of expectation give

$$\begin{aligned} \mathbb{E} \left[\|\mathbf{g}^{(k+1)} \mathbf{M}\|^2 \right] &= \mathbb{E} \left[\left\langle \sum_i \mathbf{g}_i^{(k+1)} \mathbf{M}_i, \sum_j \mathbf{g}_j^{(k+1)} \mathbf{M}_j \right\rangle \right] \\ &= \sum_{i,j} \mathbb{E}[\mathbf{g}_i^{(k+1)} \mathbf{g}_j^{(k+1)}] \langle \mathbf{M}_i, \mathbf{M}_j \rangle = \sigma^2 \sum_i \|\mathbf{M}_i\|^2 = \sigma^2 \|\mathbf{M}\|_F^2, \end{aligned} \quad (33)$$

since independence and zero means give $\mathbb{E}[\mathbf{g}_i^{(k+1)} \mathbf{g}_j^{(k+1)}] = 0$ for $i \neq j$, while $\mathbb{E}[(\mathbf{g}_i^{(k+1)})^2] = \sigma^2$ for every i .

Replacing $\mathbf{M}$ in Eq. (33) by $\mathbf{I}_+^{(k)} \mathbf{A}^{(k)} \mathbf{P}^{(k)}$ and $\mathbf{I}_-^{(k)} \mathbf{A}^{(k)} \mathbf{P}^{(k)}$, and then using Eq. (30), gives Eq. (32). $\square$

Thus, whenever $\mathbf{A}_j^{(k)} \mathbf{P}^{(k)} \neq \mathbf{0}$, the active-side projected normal has a larger expected squared length. A positive expected difference alone does not determine how often an individual comparison is correct. Therefore, we introduce the following single-point success probability estimation.

Proposition 2 (Single-point success probability). *Follow the same assumption in Pro. 1 and define $\tau := \sum_{i \in \mathbb{S}/\{j\}} \langle \mathbf{A}_i^{(k)} \mathbf{P}^{(k)}, \mathbf{A}_j^{(k)} \mathbf{P}^{(k)} \rangle^2 \geq 0$, then we have the single-point success probability,*

$$\Pr \left[\|\bar{\mathbf{g}}^{(k)}\| > \|\bar{\mathbf{g}}'^{(k)}\| \right] = \frac{1}{2} + \frac{1}{\pi} \arcsin \left(\frac{\|\mathbf{A}_j^{(k)} \mathbf{P}^{(k)}\|^2}{\sqrt{\|\mathbf{A}_j^{(k)} \mathbf{P}^{(k)}\|^4 + 4\tau}} \right) > \frac{1}{2}. \quad (34)$$

Proof. Define $\mathbf{u}_i := \mathbf{P}^{(k)} (\mathbf{A}_i^{(k)})^\top$. According to Eq. (29), $\bar{\mathbf{g}}'^{(k)} = \sum_{i \in \mathbb{S}/\{j\}} \mathbf{g}_i^{(k+1)} \mathbf{u}_i$, $\bar{\mathbf{g}}^{(k)} = \sum_{i \in \mathbb{S}/\{j\}} \mathbf{g}_i^{(k+1)} \mathbf{u}_i + \mathbf{g}_j^{(k+1)} \mathbf{u}_j$, and $\tau = \sum_{i \in \mathbb{S}/\{j\}} \langle \mathbf{u}_i, \mathbf{u}_j \rangle^2$. Eq. (31) can then be written as

$$\|\bar{\mathbf{g}}^{(k)}\|^2 - \|\bar{\mathbf{g}}'^{(k)}\|^2 = \mathbf{g}_j^{(k+1)} \left(2 \sum_{i \in \mathbb{S}/\{j\}} \mathbf{g}_i^{(k+1)} \langle \mathbf{u}_i, \mathbf{u}_j \rangle + \mathbf{g}_j^{(k+1)} \|\mathbf{u}_j\|^2 \right). \quad (35)$$

Define $Z_1 := \mathbf{g}_j^{(k+1)}$ and $Z_2 := 2 \sum_{i \in \mathbb{S}/\{j\}} \mathbf{g}_i^{(k+1)} \langle \mathbf{u}_i, \mathbf{u}_j \rangle + \mathbf{g}_j^{(k+1)} \|\mathbf{u}_j\|^2$. The active-side projected normal is therefore longer exactly when $Z_1 Z_2 > 0$, that is, when Z_1 and Z_2 have the same sign.

³ Under Kaiming initialization [14], all network weights are independent zero-mean Gaussian variables.

By assumption in Pro. 1, the coefficients $\mathbf{g}_i^{(k+1)}$ are independent zero-mean Gaussian variables and $\mathbf{u}_i$ are fixed. For any $\alpha, \beta \in \mathbb{R}$, $\alpha Z_1 + \beta Z_2$ is a linear combination of $\mathbf{g}_i^{(k+1)}$ and is therefore Gaussian. Hence, (Z_1, Z_2) is a jointly Gaussian pair. Their means are zero by the linearity of expectation.

Since $Z_1 = \mathbf{g}_j^{(k+1)}$, we immediately have $\text{Var}(Z_1) = \sigma^2$. For Z_2 , we have

$$\begin{aligned} \text{Var}(Z_2) &= \text{Var} \left(2 \sum_{i \in \mathbb{S}/\{j\}} \mathbf{g}_i^{(k+1)} \langle \mathbf{u}_i, \mathbf{u}_j \rangle + \mathbf{g}_j^{(k+1)} \|\mathbf{u}_j\|^2 \right) \\ &= 4 \sum_{i \in \mathbb{S}/\{j\}} \langle \mathbf{u}_i, \mathbf{u}_j \rangle^2 \text{Var}(\mathbf{g}_i^{(k+1)}) + \|\mathbf{u}_j\|^4 \text{Var}(\mathbf{g}_j^{(k+1)}) = \sigma^2 (4\tau + \|\mathbf{u}_j\|^4). \end{aligned} \quad (36)$$

Since Z_1 and Z_2 have zero mean, their covariance is $\text{Cov}(Z_1, Z_2) = \mathbb{E}[Z_1 Z_2]$. For $i \in \mathbb{S}/\{j\}$, the independence gives $\mathbb{E}[\mathbf{g}_j^{(k+1)} \mathbf{g}_i^{(k+1)}] = \mathbb{E}[\mathbf{g}_j^{(k+1)}] \mathbb{E}[\mathbf{g}_i^{(k+1)}] = 0$. Therefore,

$$\begin{aligned} \text{Cov}(Z_1, Z_2) &= \mathbb{E} \left[\mathbf{g}_j^{(k+1)} \left(2 \sum_{i \in \mathbb{S}/\{j\}} \mathbf{g}_i^{(k+1)} \langle \mathbf{u}_i, \mathbf{u}_j \rangle + \mathbf{g}_j^{(k+1)} \|\mathbf{u}_j\|^2 \right) \right] \\ &= 2 \sum_{i \in \mathbb{S}/\{j\}} \langle \mathbf{u}_i, \mathbf{u}_j \rangle \mathbb{E}[\mathbf{g}_j^{(k+1)} \mathbf{g}_i^{(k+1)}] + \|\mathbf{u}_j\|^2 \mathbb{E}[(\mathbf{g}_j^{(k+1)})^2] = \sigma^2 \|\mathbf{u}_j\|^2. \end{aligned} \quad (37)$$

Using $\text{Var}(Z_1) = \sigma^2$, $\text{Var}(Z_2) = \sigma^2(4\tau + \|\mathbf{u}_j\|^4)$, and $\text{Cov}(Z_1, Z_2) = \sigma^2 \|\mathbf{u}_j\|^2$, the correlation coefficient between Z_1 and Z_2 is

$$\text{Corr}(Z_1, Z_2) = \frac{\text{Cov}(Z_1, Z_2)}{\sqrt{\text{Var}(Z_1) \text{Var}(Z_2)}} = \frac{\|\mathbf{u}_j\|^2}{\sqrt{\|\mathbf{u}_j\|^4 + 4\tau}}. \quad (38)$$

Since $\mathbf{u}_j \neq \mathbf{0}$, $\text{Corr}(Z_1, Z_2) > 0$. Thus, Z_1 and Z_2 are positively correlated.

Dividing Z_1 and Z_2 by their positive standard deviations does not change their signs or their correlation coefficient. The resulting variables form a standard jointly Gaussian pair. For such a pair, the standard Gaussian quadrant identity gives $\Pr[Z_1 > 0, Z_2 > 0] = 1/4 + \arcsin(\text{Corr}(Z_1, Z_2))/(2\pi)$. Moreover, the zero-mean jointly Gaussian distribution is centrally symmetric, so $\Pr[Z_1 < 0, Z_2 < 0] = \Pr[Z_1 > 0, Z_2 > 0]$. By Eq. (35), the active-side projected normal is longer exactly when $Z_1 Z_2 > 0$. Therefore,

$$\begin{aligned} \Pr \left[\|\bar{\mathbf{g}}^{(k)}\| > \|\bar{\mathbf{g}}'^{(k)}\| \right] &= \Pr[Z_1 Z_2 > 0] = \frac{1}{2} + \frac{1}{\pi} \arcsin(\text{Corr}(Z_1, Z_2)) \\ &= \frac{1}{2} + \frac{1}{\pi} \arcsin \left(\frac{\|\mathbf{u}_j\|^2}{\sqrt{\|\mathbf{u}_j\|^4 + 4\tau}} \right). \end{aligned} \quad (39)$$

Since $\text{Corr}(Z_1, Z_2) > 0$, the probability in Eq. (39) is strictly greater than 1/2. Finally, substituting $\mathbf{u}_j = \mathbf{P}^{(k)}(\mathbf{A}_j^{(k)})^\top$ proves Eq. (34). $\square$

Proposition 3. *The $\tau := \sum_{i \in \mathbb{S}/\{j\}} \langle \mathbf{A}_i^{(k)} \mathbf{P}^{(k)}, \mathbf{A}_j^{(k)} \mathbf{P}^{(k)} \rangle^2$ is the sum of the squared inner products between the projected weight of the target neuron (neuron*

j) and the projected weights of the other active neurons (neuron $i \in \mathbb{S}/\{j\}$). It therefore measures their total alignment with the target direction $\mathbf{A}_j^{(k)} \mathbf{P}^{(k)}$. A larger alignment τ generally reduces the single-point success probability.

4.3 Normal Alignment in Hard-Label Setting

The preceding analysis gives the length comparison: according to Eq. (34), the side with the longer $\bar{\mathbf{g}}^{(k)}$ is predicted to be the target active side with a higher probability. However, the hard-label queries recover only the unit projected normals $\bar{\mathbf{n}}^{(k)}$ and $\bar{\mathbf{n}}'^{(k)}$ as stated in the last paragraph of Sect. 4.1, thereby losing the lengths of $\bar{\mathbf{g}}^{(k)}$ and $\bar{\mathbf{g}}'^{(k)}$. Consequently, the length comparison cannot be applied directly. We therefore compare the absolute alignments of the two unit projected normals with the recovered target weight $\hat{\mathbf{A}}_j^{(k)}$.

Proposition 4 (Equivalence of the length and alignment comparisons).

Assume $\bar{\mathbf{g}}^{(k)} \neq \mathbf{0}$, $\bar{\mathbf{g}}'^{(k)} \neq \mathbf{0}$, $\mathbf{P}^{(k)}(\hat{\mathbf{A}}_j^{(k)})^\top \neq \mathbf{0}$, and $\mathbf{P}^{(k)}(\hat{\mathbf{A}}_j^{(k)})^\top \nparallel \bar{\mathbf{g}}'^{(k)}$. Then

$$\|\bar{\mathbf{g}}^{(k)}\| > \|\bar{\mathbf{g}}'^{(k)}\| \iff \left| \left\langle \bar{\mathbf{n}}^{(k)}, (\hat{\mathbf{A}}_j^{(k)})^\top \right\rangle \right| > \left| \left\langle \bar{\mathbf{n}}'^{(k)}, (\hat{\mathbf{A}}_j^{(k)})^\top \right\rangle \right|. \quad (40)$$

Proof. According to Eq. (26), we have $\mathbf{P}^{(k)}\bar{\mathbf{n}}^{(k)} = \bar{\mathbf{n}}^{(k)}$ and $\mathbf{P}^{(k)}\bar{\mathbf{n}}'^{(k)} = \bar{\mathbf{n}}'^{(k)}$. Define $\mathbf{m} := \mathbf{P}^{(k)}(\hat{\mathbf{A}}_j^{(k)})^\top / \|\mathbf{P}^{(k)}(\hat{\mathbf{A}}_j^{(k)})^\top\|$. Using these identities and the symmetry of $\mathbf{P}^{(k)}$, we obtain

$$\begin{aligned} \left\langle \bar{\mathbf{n}}^{(k)}, (\hat{\mathbf{A}}_j^{(k)})^\top \right\rangle &= \left\langle \mathbf{P}^{(k)}\bar{\mathbf{n}}^{(k)}, (\hat{\mathbf{A}}_j^{(k)})^\top \right\rangle = \left\langle \bar{\mathbf{n}}^{(k)}, \mathbf{P}^{(k)}(\hat{\mathbf{A}}_j^{(k)})^\top \right\rangle = \left\| \mathbf{P}^{(k)}(\hat{\mathbf{A}}_j^{(k)})^\top \right\| \left\langle \bar{\mathbf{n}}^{(k)}, \mathbf{m} \right\rangle, \\ \left\langle \bar{\mathbf{n}}'^{(k)}, (\hat{\mathbf{A}}_j^{(k)})^\top \right\rangle &= \left\langle \mathbf{P}^{(k)}\bar{\mathbf{n}}'^{(k)}, (\hat{\mathbf{A}}_j^{(k)})^\top \right\rangle = \left\langle \bar{\mathbf{n}}'^{(k)}, \mathbf{P}^{(k)}(\hat{\mathbf{A}}_j^{(k)})^\top \right\rangle = \left\| \mathbf{P}^{(k)}(\hat{\mathbf{A}}_j^{(k)})^\top \right\| \left\langle \bar{\mathbf{n}}'^{(k)}, \mathbf{m} \right\rangle. \end{aligned} \quad (41)$$

The common factor $\|\mathbf{P}^{(k)}(\hat{\mathbf{A}}_j^{(k)})^\top\|$ is positive and therefore does not affect the ordering of the absolute inner products.

Since $\hat{\mathbf{A}}_j^{(k)}$ and $\hat{\mathbf{A}}_j'^{(k)}$ differ only by a nonzero scalar, Eq. (29) shows that $\bar{\mathbf{g}}^{(k)} - \bar{\mathbf{g}}'^{(k)}$ is parallel to $\mathbf{m}$. Hence, there exist $\alpha, \beta \in \mathbb{R}$ and a vector $\mathbf{m}_\perp \perp \mathbf{m}$ such that $\bar{\mathbf{g}}'^{(k)} = \alpha \mathbf{m} + \mathbf{m}_\perp$ and $\bar{\mathbf{g}}^{(k)} = \beta \mathbf{m} + \mathbf{m}_\perp$. Thus, they share the same orthogonal component $\mathbf{m}_\perp$. Since $\mathbf{m}$ is a unit vector and $\mathbf{m}_\perp \perp \mathbf{m}$, we have $\|\bar{\mathbf{g}}'^{(k)}\|^2 = \alpha^2 + \|\mathbf{m}_\perp\|^2$ and $\|\bar{\mathbf{g}}^{(k)}\|^2 = \beta^2 + \|\mathbf{m}_\perp\|^2$. Using $\bar{\mathbf{n}}^{(k)} = \bar{\mathbf{g}}^{(k)} / \|\bar{\mathbf{g}}^{(k)}\|$ and $\bar{\mathbf{n}}'^{(k)} = \bar{\mathbf{g}}'^{(k)} / \|\bar{\mathbf{g}}'^{(k)}\|$, we obtain $|\langle \bar{\mathbf{n}}^{(k)}, \mathbf{m} \rangle|^2 = \beta^2 / (\beta^2 + \|\mathbf{m}_\perp\|^2)$ and $|\langle \bar{\mathbf{n}}'^{(k)}, \mathbf{m} \rangle|^2 = \alpha^2 / (\alpha^2 + \|\mathbf{m}_\perp\|^2)$. Consequently,

$$\begin{aligned} & \left| \left\langle \bar{\mathbf{n}}^{(k)}, \mathbf{m} \right\rangle \right|^2 - \left| \left\langle \bar{\mathbf{n}}'^{(k)}, \mathbf{m} \right\rangle \right|^2 \\ &= \frac{\beta^2}{\beta^2 + \|\mathbf{m}_\perp\|^2} - \frac{\alpha^2}{\alpha^2 + \|\mathbf{m}_\perp\|^2} = \frac{\beta^2(\alpha^2 + \|\mathbf{m}_\perp\|^2) - \alpha^2(\beta^2 + \|\mathbf{m}_\perp\|^2)}{(\beta^2 + \|\mathbf{m}_\perp\|^2)(\alpha^2 + \|\mathbf{m}_\perp\|^2)} \\ &= \frac{\|\mathbf{m}_\perp\|^2(\beta^2 - \alpha^2)}{(\beta^2 + \|\mathbf{m}_\perp\|^2)(\alpha^2 + \|\mathbf{m}_\perp\|^2)} = \frac{\|\mathbf{m}_\perp\|^2}{\|\bar{\mathbf{g}}^{(k)}\|^2 \|\bar{\mathbf{g}}'^{(k)}\|^2} \left(\|\bar{\mathbf{g}}^{(k)}\|^2 - \|\bar{\mathbf{g}}'^{(k)}\|^2 \right). \end{aligned} \quad (42)$$

With the assumptions $\bar{\mathbf{g}}^{(k)} \neq \mathbf{0}$, $\bar{\mathbf{g}}'^{(k)} \neq \mathbf{0}$, $\mathbf{P}^{(k)}(\hat{\mathbf{A}}_j^{(k)})^\top \neq \mathbf{0}$, and $\mathbf{P}^{(k)}(\hat{\mathbf{A}}_j^{(k)})^\top \nparallel \bar{\mathbf{g}}'^{(k)}$, the factor $\|\mathbf{m}_\perp\|^2 / (\|\bar{\mathbf{g}}^{(k)}\|^2 \|\bar{\mathbf{g}}'^{(k)}\|^2)$ is strictly positive. Hence, $\|\bar{\mathbf{g}}^{(k)}\| >$

$\|\bar{\mathbf{g}}^{(k)}\|$ if and only if $|\langle \bar{\mathbf{n}}^{(k)}, \mathbf{m} \rangle| > |\langle \bar{\mathbf{n}}'^{(k)}, \mathbf{m} \rangle|$. By Eq. (41), the latter is equivalent to $|\langle \bar{\mathbf{n}}^{(k)}, (\hat{\mathbf{A}}_j^{(k)})^\top \rangle| > |\langle \bar{\mathbf{n}}'^{(k)}, (\hat{\mathbf{A}}_j^{(k)})^\top \rangle|$, which proves Pro. 4. $\square$

According to Pro. 4, the alignment comparison has the same single-point success probability given in Eq. (34). Based on this comparison, we introduce the sign recovery method *Normal Alignment* for hard-label networks.

Proposition 5 (Normal Alignment). *At each dual point, the side whose unit projected decision-facet normal $\bar{\mathbf{n}}^{(k)}$ has the larger absolute inner product with $(\hat{\mathbf{A}}_j^{(k)})^\top$ is predicted to be the target active side. If this prediction agrees with the active side indicated by the recovered signature, the dual point votes to retain the sign of $\hat{\mathbf{A}}_j^{(k)}$; otherwise, it votes to reverse its sign.*

Normal Alignment repeats this comparison at multiple dual points and aggregates the resulting votes. The majority vote determines whether the sign of the recovered signature is retained or reversed. The fraction of valid votes supporting this decision is used as its confidence level. For a target neuron, let n_+ and n_- denote the numbers of votes for retaining and reversing the recovered signature, respectively, and let $n_{\text{valid}} := n_+ + n_-$. For $n_{\text{valid}} > 0$, define the confidence level as $\alpha := \max\{n_+, n_-\}/n_{\text{valid}}$. Given a confidence threshold $\alpha_0 \in (1/2, 1]$, the *Normal Alignment* retains the recovered signature if $n_+ > n_-$ and $\alpha \geq \alpha_0$, and reverses it if $n_- > n_+$ and $\alpha \geq \alpha_0$. Otherwise, the available votes are insufficient to determine the sign, which remains unresolved.

White-Box Validation of the Normal Alignment. We experimentally validate the *Normal Alignment* in the white-box setting on the CIFAR-10 DNNs with architectures $192\text{-}d \times 3\text{-}10$ for $d \in \{32, 64, 128, 256\}$. As shown in Fig. 3 in Supp. A, $n_{\text{attempt}} = 200$ provides high sign recovery accuracy. Table 3 summarizes the layer-wise results. For each dual point, the projected normal length comparison in Eq. (31) is quantified by $\frac{\|\bar{\mathbf{g}}^{(k)}\|}{\|\bar{\mathbf{g}}'^{(k)}\|}$. The corresponding column reports the median of this ratio over all evaluated dual points in each layer. For each neuron in a layer, Eq. (34) is used to compute a theoretical success probability at each of its dual points and “ p_{th} ” is the mean of these probabilities over all evaluated dual points in the layer. Correspondingly, p_{obs} is the proportion of correct single-point votes among all evaluated dual points in that layer. The column of “Agreement” reports the percentage of evaluated dual points in each layer satisfying the equivalence in Eq. (40) of Pro. 4. The column of “Signs recovered” gives the number of correctly recovered neuron signs in each layer after aggregating 200 votes per neuron.

As shown in Table 3, the median of the ratio $\frac{\|\bar{\mathbf{g}}^{(k)}\|}{\|\bar{\mathbf{g}}'^{(k)}\|}$ is greater than one in all twelve hidden layers. For all the evaluated dual points of the four DNNs, the mean single-point success probabilities are $p_{\text{th}} = 66.01\%$ theoretically and $p_{\text{obs}} = 70.12\%$ empirically. Thus, the theoretical model captures the advantage over random guessing, although it underestimates its magnitude. Consistent with Pro. 4, the length and alignment rules agree (Eq. (40) is satisfied) on

Table 3: Layer-wise white-box validation of *Normal Alignment* on CIFAR-10 DNNs 192- d - d - d -10 using $n_{\text{attempt}} = 200$ dual points per neuron.

d	Layer k	$\ \bar{\mathbf{g}}^{(k)}\ /\ \bar{\mathbf{g}}'^{(k)}\ $	$p_{\text{th}}/p_{\text{obs}}$ (%)	Agreement (%)	Signs recovered
32	1	1.014	74.94/76.61	100.00	32/32
	2	1.010	64.31/66.92	100.00	30/32
	3	1.016	66.86/69.56	100.00	31/32
64	1	1.008	71.04/74.68	100.00	64/64
	2	1.007	65.12/69.12	100.00	64/64
	3	1.012	66.25/71.84	100.00	62/64
128	1	1.004	68.04/73.15	100.00	128/128
	2	1.004	65.62/68.79	100.00	128/128
	3	1.004	65.40/68.82	100.00	126/128
256	1	1.002	64.99/70.81	100.00	256/256
	2	1.002	66.00/68.95	100.00	255/256
	3	1.002	64.41/68.73	100.00	241/256

every evaluated dual point. After vote aggregation, *Normal Alignment* correctly recovers 1417 of the 1440 neuron signs, giving an aggregate recovery accuracy of $\frac{1417}{1440} \times 100\% = 98.40\%$. In particular, all signs in the first hidden layer are recovered correctly for all four models. Meanwhile, the second and third hidden layers contain 20 incorrectly recovered signs and 3 tied outcomes.

5 *eSOE+Alignment*: Combining *Normal Alignment* with Hard-Label *SOE*

At NeurIPS 2024, Foerster *et al.* [13] empirically observed that many neuron signs recovered by *Neuron Wiggle* [4] remained at low confidence and that collecting additional critical points did not improve their confidence. Therefore, they performed an exhaustive search on these low-confidence signs, which led to a significant increase in the number of model queries and runtime, even turning the so-called polynomial-time attack into an exponential-time attack. As a probabilistic voting method, the *Normal Alignment* may also leave some neuron signs undetermined when their voting confidence is insufficient. In contrast, Hard-label *SOE* [3] can deterministically recover the activation states of all neurons in layer k , but it requires $\text{rank}(\hat{\mathbf{A}}^{(k)} \mathbf{F}^{(k-1)}) = d^{(k+1)}$ to solve the linear system in Eq. (21). Inspired by the raw-output *SOE+Wiggle* [18], we proposed hard-label *eSOE+Alignment* for high-confidence sign recovery.

Recall from Sect. 3.2, *SOE + Wiggle* contains two main parts: removing inactive neurons identified with high-confidence level in Eq. (16) and extending the system from multiple points in Eq. (18). In S1 access, removing inactive neurons remains straightforward: at a selected transition point, the signs recovered by *Normal Alignment* with high confidence are used to identify inactive neurons, and their corresponding zero entries are removed from the hard-label *SOE* Eq. (21) in Sect. 3.2. The system extension, however, cannot be applied directly.

Finding Compatible Transition Points. According to Eq. (21), the unknown vector is $\left((\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)})\mathbf{I}^{(k)}\mathbf{S}^{(k)}\right)^\top$. To extend the linear system in Eq. (21), the selected points must share the same unknowns. Therefore, two conditions should be simultaneously satisfied:

- First, the selected points must share the activation states in layer k and all subsequent layers; otherwise, their corresponding hard-label *SOE* systems in Eq. (21) have different unknown vectors;
- Second, the points should remain on the decision boundary between the same two classes, *i.e.*, $\mathbb{D}_{ab}$.

We can only walk along the decision boundary to ensure that no future neurons have toggled, and check the output label from slightly perturbing $\mathcal{F}_\theta(\mathbf{x}_t + \boldsymbol{\delta}^{(1)})$ and $\mathcal{F}_\theta(\mathbf{x}_t - \boldsymbol{\delta}^{(1)})$ to keep $\mathbf{x}_t \in \mathbb{D}_{ab}$. We use the idea for locating dual points introduced in [5] to find compatible transition points on the decision boundary between fixed two classes, specifically,

- *Step 1:* From $\mathbf{x}_1 \in \mathbb{D}_{ab} \cap \mathbb{L}_{\mathbf{x}_1}$, the attacker makes a random excursion and uses hard-label binary search to relocate another point $\mathbf{x}'_1 \in \mathbb{D}_{ab} \cap \mathbb{L}_{\mathbf{x}_1}$. Then the difference $\mathbf{x}'_1 - \mathbf{x}_1$ determines a direction along $\mathbb{D}_{ab} \cap \mathbb{L}_{\mathbf{x}_1}$. Following the direction of $\mathbf{x}'_1 - \mathbf{x}_1$, the attacker walks until the decision boundary bends at a dual point $\mathbf{x}_2$ (suppose that $\mathbf{x}_1 \in \mathbb{D}_{ab} \cap \mathbb{L}_{\mathbf{x}_2}$).
- *Step 2:* Since the parameters of layers $1, \dots, k-1$ have already been recovered, the attacker can evaluate the pre-activation values of the neurons in these layers at $\mathbf{x}_2$. If a neuron in a preceding layer is zero before ReLU, the bend is attributed to that layer; the attacker then makes a random excursion and uses binary search to locate another $\mathbf{x}'_2$ on the adjacent decision facet $\mathbb{D}_{ab} \cap \mathbb{L}'_{\mathbf{x}_2}$. Otherwise, the bend may be caused by a neuron in layer k or a subsequent layer, so the attacker terminates the current search path.
- *Step 3:* If the bend is attributed to a neuron in a preceding layer, the attacker then collects $\mathbf{x}'_2$ for *SOE* extension and continues to find the next decision boundary bend along the direction of $\mathbf{x}'_2 - \mathbf{x}_2$.

Repeating this procedure from the starting point $\mathbf{x}_1$ yields sufficient transition points for the following extension.

Scale-Free Hard-label *SOE* Extension. Let $\mathbf{x}_1, \dots, \mathbf{x}_T$ be the collected transition points. Since these points share the same activation matrices in layer k and all subsequent layers, their $\mathbf{I}^{(k)}$ and $\mathbf{G}^{(k+1)}$ are identical. Let $\mathbf{g}_t^{(1)}$ be the normal of the decision facet $\mathbb{D}_{ab} \cap \mathbb{L}_{\mathbf{x}_t} \subset \mathbb{R}^{d^{(1)}}$, and $\mathbf{n}_t^{(1)} = \mathbf{g}_t^{(1)} / \|\mathbf{g}_t^{(1)}\|$. The decision-facet normal is often already available from signature recovery and can therefore be reused here, avoiding the additional oracle queries needed to search for perturbations and construct the equations in Eq. (21). According to Eq. (24) in Sect. 4.1, $\mathbf{g}_t^{(1)} = (\mathbf{A}^{(k)}\mathbf{F}_{\mathbf{x}_t}^{k-1})^\top \mathbf{I}^{(k)}(\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)})^\top$. With the recovered

signature $\hat{\mathbf{A}}^{(k)}$ and prefix map $\mathbf{F}_{\mathbf{x}_t}^{(k-1)}$, we have

$$\mathbf{n}_t^{(1)} = \left(\hat{\mathbf{A}}^{(k)} \mathbf{F}_{\mathbf{x}_t}^{(k-1)} \right)^\top \frac{\mathbf{S}^{(k)}}{\|\mathbf{g}_t^{(1)}\|} \left[\mathbf{I}^{(k)} \left(\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)} \right)^\top \right]. \quad (43)$$

Although $\mathbf{S}^{(k)} \mathbf{I}^{(k)} (\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)})^\top$ is common to all selected points, the $1/\|\mathbf{g}_t^{(1)}\|$ generally differs across them. Therefore, these hard-label *SOE* systems cannot be combined directly. Since $\mathbf{n}_t^{(1)}$ is a unit vector, $(\mathbf{Id}_{d^{(1)}} - \mathbf{n}_t^{(1)} \mathbf{n}_t^{(1)\top}) \mathbf{n}_t^{(1)} = \mathbf{0}$, where $\mathbf{Id}_{d^{(1)}}$ is the $d^{(1)} \times d^{(1)}$ identity matrix. Multiplying both sides of Eq. (43) by $\mathbf{Id}_{d^{(1)}} - \mathbf{n}_t^{(1)} \mathbf{n}_t^{(1)\top}$ to remove $1/\|\mathbf{g}_t^{(1)}\|$, and it gives

$$(\mathbf{Id}_{d^{(1)}} - \mathbf{n}_t^{(1)} \mathbf{n}_t^{(1)\top}) \left(\hat{\mathbf{A}}^{(k)} \mathbf{F}_{\mathbf{x}_t}^{(k-1)} \right)^\top \mathbf{S}^{(k)} \left[\mathbf{I}^{(k)} \left(\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)} \right)^\top \right] = \mathbf{0}. \quad (44)$$

Now, the unknown vector in Eq. (44) is the same for all selected transition points.

Again, let $\mathbb{K}$ contain the neurons that *Normal Alignment* identifies as inactive in layer k at $\mathbf{x}_t$ with high confidence, and $\mathbb{U} := [d^{(k+1)}] \setminus \mathbb{K}$ contain the remaining neurons. For a vector $\mathbf{v}$, $[\mathbf{v}]_{\mathbb{U}}$ denotes the subvector indexed by $\mathbb{U}$. Restricting Eq. (44) to $\mathbb{U}$ and combining the equations from all collected transition points, we obtain

$$\begin{bmatrix} [(\mathbf{Id}_{d^{(1)}} - \mathbf{n}_1^{(1)} \mathbf{n}_1^{(1)\top}) (\hat{\mathbf{A}}^{(k)} \mathbf{F}_{\mathbf{x}_1}^{(k-1)})^\top]_{\mathbb{U}} \\ [(\mathbf{Id}_{d^{(1)}} - \mathbf{n}_2^{(1)} \mathbf{n}_2^{(1)\top}) (\hat{\mathbf{A}}^{(k)} \mathbf{F}_{\mathbf{x}_2}^{(k-1)})^\top]_{\mathbb{U}} \\ \vdots \\ [(\mathbf{Id}_{d^{(1)}} - \mathbf{n}_T^{(1)} \mathbf{n}_T^{(1)\top}) (\hat{\mathbf{A}}^{(k)} \mathbf{F}_{\mathbf{x}_T}^{(k-1)})^\top]_{\mathbb{U}} \end{bmatrix} \left[\mathbf{S}^{(k)} \mathbf{I}^{(k)} \left(\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)} \right)^\top \right]_{\mathbb{U}} = \mathbf{0}. \quad (45)$$

Proposition 6 (*eSOE+Alignment*). *Assume every neuron in $\mathbb{K}$ is inactive at the selected transition points and the entries of $[(\mathbf{G}_a^{(k+1)} - \mathbf{G}_b^{(k+1)})^\top]_{\mathbb{U}}$ corresponding to active neurons are nonzero. If the coefficient matrix in Eq. (45) has rank $|\mathbb{U}| - 1$, then its nonzero solution is uniquely determined up to a scalar multiple. Then, *eSOE+Alignment* recovers all the neuron signs in layer k in polynomial time.*

The correctness of the sign assignment returned by *eSOE+Alignment* can be assessed by attempting signature recovery for the next layer, as proposed by [13]. If the next-layer signatures cannot be recovered, *eSOE* may have incorrectly eliminated a column corresponding to a neuron that is actually active at the selected transition points, making the solution unreliable. We therefore discard the signs returned by *eSOE+Alignment* and instead use the signs predicted by *Normal Alignment*. We then apply the exhaustive-search strategy of [13] to enumerate candidate assignments for the low-confidence signs until the next-layer signatures are successfully recovered. We handle *eSOE+Toggle* analogously: when its

reduced *SOE* solution is unreliable, we apply the same exhaustive-search strategy to the low-confidence signs predicted by *Future Toggle*.

We validate *eSOE+Alignment* on the CIFAR-10 model with architecture 3072-256×3-64-10. Following the proof-of-concept setting in [5], we use exact decision-facet normals and set $n_{\text{attempt}} = 200$. For every hidden layer, the projected coefficient matrix satisfies the rank condition in Proposition 6, and *eSOE+Alignment* correctly recovers all 832 signs, as reported in Table 4 in Supp. A. The same result is observed in Table 1, where *eSOE+Alignment* recovers all 512 and 320 signs, respectively. Thus, no fallback exhaustive search is required in any of the evaluated settings. For the 3072-256×3-64-10 model, complete sign recovery takes about 1h43m on our 8-core CPU without neuron-level parallelism. For reference, Carlini *et al.* [5] reported an estimated runtime of 8.5 hours for complete sign recovery using *Future Toggle* on a 256-core server, with 64 neurons processed in parallel.

6 Experiments

We evaluate sign recovery on two trained ReLU DNNs: a CIFAR-10 model with architecture 192-64×8-10 and an MNIST model with architecture 64-96×3-32-10, containing 42,122 and 28,298 parameters, respectively.⁴ Both networks use ReLU activations and Kaiming normal initialization [14]. For the target layer k , the sign recoveries are evaluated assuming that the preceding layers have been recovered and the layer k 's neuron signatures are known up to sign.

Following the experimental convention of [5], we assume that the dual points have been precomputed because this step is shared by signature recovery and sign recovery. We use 200 attempted dual points per neuron for *Normal Alignment* and *eSOE+Alignment*, and evaluate *Future Toggle* and *eSOE+Toggle* with budgets of both 200 and 1000. For CIFAR-10, we follow the proof-of-concept setting of [5]. Decision-facet normals are computed directly from model parameters. *Future Toggle* additionally uses white-box information to walk along the decision boundary and handle non-future toggles. For MNIST, we recover the unit decision-facet normals using the procedure described in Sect. 4.1, and *Future Toggle* discards a dual point if the boundary walk first encounters a non-future toggle. Tables 5, 6, and 7 in Supp. A summarize the results across all hidden layers. *eSOE+Alignment* correctly recovers all 512 and 320 signs in the two models, respectively.

7 Conclusion

This paper proposes *Normal Alignment*, an improved statistical sign-recovery method for hard-label ReLU network extraction to address the limitations of *Future Toggle*. Using projected decision-facet normals at dual points, it achieves

⁴ CIFAR-10 and MNIST images are resized and flattened to $8 \times 8 \times 3 = 192$ and $8 \times 8 = 64$ dimensions, respectively.

higher voting accuracy and pushes errors to low-confidence ranks. Combined with extended hard-label *SOE*, *eSOE+Alignment* achieves exact polynomial-time full sign recovery without exponential enumeration. Evaluations on CIFAR-10 and MNIST confirm its superiority.

References

1. Asselineau, R., Derbez, P., Fouque, P., Minaud, B.: Cryptanalytic extraction of deep neural networks with non-linear activations. In: Heninger, N., Rosulek, M. (eds.) *Advances in Cryptology - CRYPTO 2026 - 46th Annual International Cryptology Conference*, Santa Barbara, CA, USA, August 17-20, 2026, *Proceedings*, Part VII. *Lecture Notes in Computer Science*, vol. 16806, pp. 36–66. Springer (2026). https://doi.org/10.1007/978-3-032-35415-0_2, https://doi.org/10.1007/978-3-032-35415-0_2
2. Batina, L., Bhasin, S., Jap, D., Picek, S.: CSI NN: Reverse engineering of neural network architectures through electromagnetic side channel. In: 28th USENIX Security Symposium (USENIX Security 19). pp. 515–532 (2019)
3. Canales-Martínez, I.A., Santos, D.: Extracting some layers of deep neural networks in the hard-label setting. In: Escudero, D., Damgård, I. (eds.) *Progress in Cryptology - LATINCRYPT 2025 - 9th International Conference on Cryptology and Information Security in Latin America*, Medellín, Colombia, October 1-3, 2025, *Proceedings*. *Lecture Notes in Computer Science*, vol. 16129, pp. 399–421. Springer (2025). https://doi.org/10.1007/978-3-032-06754-8_15, https://doi.org/10.1007/978-3-032-06754-8_15
4. Canales-Martínez, I.A., Chávez-Saab, J., Hambitzer, A., Rodríguez-Henríquez, F., Satpute, N., Shamir, A.: Polynomial time cryptanalytic extraction of neural network models. In: Joye, M., Leander, G. (eds.) *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Zurich, Switzerland, May 26-30, 2024, *Proceedings*, Part III. *Lecture Notes in Computer Science*, vol. 14653, pp. 3–33. Springer (2024). https://doi.org/10.1007/978-3-031-58734-4_1, https://doi.org/10.1007/978-3-031-58734-4_1
5. Carlini, N., Chávez-Saab, J., Hambitzer, A., Rodríguez-Henríquez, F., Shamir, A.: Polynomial time cryptanalytic extraction of deep neural networks in the hard-label setting. In: Fehr, S., Fouque, P. (eds.) *Advances in Cryptology - EUROCRYPT 2025 - 44th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Madrid, Spain, May 4-8, 2025, *Proceedings*, Part I. pp. 364–396. *Lecture Notes in Computer Science*, Springer (2025). https://doi.org/10.1007/978-3-031-91107-1_13, https://doi.org/10.1007/978-3-031-91107-1_13
6. Carlini, N., Jagielski, M., Mironov, I.: Cryptanalytic extraction of neural network models. In: Micciancio, D., Ristenpart, T. (eds.) *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference*, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, *Proceedings*, Part III. pp. 189–218. *Lecture Notes in Computer Science*, Springer (2020). https://doi.org/10.1007/978-3-030-56877-1_7, https://doi.org/10.1007/978-3-030-56877-1_7
7. Chen, Y., Dong, X., Guo, J., Shen, Y., Wang, A., Wang, X.: Hard-label cryptanalytic extraction of neural network models. In: Chung, K., Sasaki, Y. (eds.)

- Advances in Cryptology - ASIACRYPT 2024 - 30th International Conference on the Theory and Application of Cryptology and Information Security, Kolkata, India, December 9-13, 2024, Proceedings, Part VIII. pp. 207–236. Lecture Notes in Computer Science, Springer (2024). https://doi.org/10.1007/978-981-96-0944-4_7, https://doi.org/10.1007/978-981-96-0944-4_7
8. Chen, Y., Dong, X., Ma, R., Shen, Y., Wang, A., Yu, H., Wang, X.: Delving into cryptanalytic extraction of prelu neural networks. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 576–607. Springer (2025)
 9. Chen, Z., Tang, S., Gao, Z., Su, Y., Qin, L., Dong, X.: Algebraic attack on convolutional neural networks with max pooling. In: Heninger, N., Rosulek, M. (eds.) Advances in Cryptology - CRYPTO 2026 - 46th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-20, 2026, Proceedings, Part VII. Lecture Notes in Computer Science, vol. 16806, pp. 3–35. Springer (2026). https://doi.org/10.1007/978-3-032-35415-0_1, https://doi.org/10.1007/978-3-032-35415-0_1
 10. Chen, Z., Tang, S., Gao, Z., Su, Y., Qin, L., Dong, X.: Algebraic cryptanalytic extraction on hard-label neural networks. Cryptology ePrint Archive, Paper 2026/1164 (2026), <https://eprint.iacr.org/2026/1164>
 11. Esteva, A., Kuprel, B., Novoa, R.A., Ko, J., Swetter, S.M., Blau, H.M., Thrun, S.: Dermatologist-level classification of skin cancer with deep neural networks. *Nature* **542**(7639), 115–118 (2017). <https://doi.org/10.1038/nature21056>, <https://doi.org/10.1038/nature21056>
 12. Fefferman, C., et al.: Reconstructing a neural net from its output. *Revista Matemática Iberoamericana* **10**(3), 507–556 (1994)
 13. Foerster, H., Mullins, R., Shumailov, I., Hayes, J.: Beyond slow signs in high-fidelity model extraction. In: Globersons, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J.M., Zhang, C. (eds.) Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024 (2024), http://papers.nips.cc/paper_files/paper/2024/hash/22ae669a35bb9e70eb93ab77c1eff5b4-Abstract-Conference.html
 14. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: 2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015. pp. 1026–1034. IEEE Computer Society (2015). <https://doi.org/10.1109/ICCV.2015.123>, <https://doi.org/10.1109/ICCV.2015.123>
 15. Ito, A., Miura, T., Todo, Y.: Is the hard-label cryptanalytic model extraction really polynomial? In: Heninger, N., Rosulek, M. (eds.) Advances in Cryptology - CRYPTO 2026 - 46th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-20, 2026, Proceedings, Part VII. Lecture Notes in Computer Science, vol. 16806, pp. 67–98. Springer (2026). https://doi.org/10.1007/978-3-032-35415-0_3, https://doi.org/10.1007/978-3-032-35415-0_3
 16. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Pereira, F., Burges, C., Bottou, L., Weinberger, K. (eds.) Advances in Neural Information Processing Systems. vol. 25. Curran Associates, Inc. (2012), https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

17. Liu, H., Siproudhis, A., Boura, C., Peyrin, T.: Model extraction of convolutional neural networks with max-pooling. *Cryptology ePrint Archive* (2026)
18. Liu, H., Siproudhis, A., Experton, S., Lorenz, P., Boura, C., Peyrin, T.: Navigating the deep: End-to-end extraction on deep neural networks. In: Daemen, J., Thomé, E. (eds.) *Advances in Cryptology - EUROCRYPT 2026 - 45th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Rome, Italy, May 10-14, 2026, *Proceedings, Part VI. Lecture Notes in Computer Science*, vol. 16546, pp. 482–512. Springer (2026). https://doi.org/10.1007/978-3-032-25333-0_17, https://doi.org/10.1007/978-3-032-25333-0_17
19. Lowd, D., Meek, C.: Adversarial learning. In: *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. pp. 641–647 (2005)
20. Oliynyk, D., Mayer, R., Rauber, A.: I know what you trained last summer: A survey on stealing machine learning models and defences. *ACM Computing Surveys* **55**(14s), 1–41 (2023)
21. Qi, X., Lei, H., Wei, L., Sun, X., Wang, M.: Cryptanalytic extraction of neural networks with various activation functions. *Cryptology ePrint Archive* (2026)
22. Sun, X., Lei, H., Wei, L., Qi, X., Hu, K., Wang, M., Wang, W.: Cryptanalytic extraction of convolutional neural networks. *Cryptology ePrint Archive*, Paper 2026/139 (2026), <https://eprint.iacr.org/2026/139>
23. Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., Ristenpart, T.: Stealing machine learning models via prediction APIs. In: *25th USENIX security symposium (USENIX Security 16)*. pp. 601–618 (2016)
24. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: *Advances in Neural Information Processing Systems 30*. pp. 5998–6008. Curran Associates, Inc. (2017), <https://proceedings.neurips.cc/paper/7181-attention-is-all-you-need>
25. Wei, L., Lei, H., Qi, X., Sun, X., Gao, L., Hu, K., Wang, W., Wang, M.: Cryptanalytic extraction of recurrent neural network models. *Cryptology ePrint Archive* (2026)

Supplementary Material

A Supporting Experimental Results

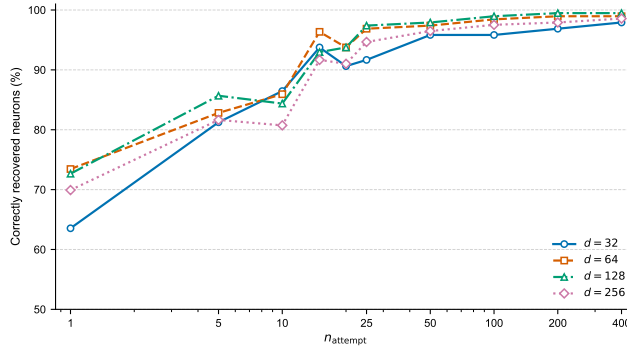


Fig. 3: Sign recovery accuracy of *Normal Alignment*. Each curve reports the percentage of correctly recovered signs across all three hidden layers after aggregating the n_{attempt} votes for each neuron.

Table 4: Sign recovery results of *Normal Alignment* and *eSOE+Alignment* on the CIFAR-10 model with architecture 3072-256 $\times$ 3-64-10.

Method	Metric	L1	L2	L3	L4	All hidden layers
<i>Normal Alignment</i>	Correct signs	256/256	256/256	255/256	63/64	830/832
	Vote accuracy p (%)	77.85	72.09	67.85	74.46	72.74
<i>eSOE+Alignment</i>	Correct signs	256/256	256/256	256/256	64/64	832/832
	Projected rank	255	146	64	63	–
	Min confidence	N/A	0.73	0.60	N/A	0.60

Projected rank: The rank of the coefficient matrix in Eq. (45) after projection and column elimination. In every layer, the reported rank equals $|\mathbb{U}| - 1$, so the nonzero solution is uniquely determined up to a scalar multiple.

Min confidence: The minimum *Normal Alignment* confidence used to eliminate inactive-neuron columns. N/A indicates that the unprojected stacked coefficient matrix already has full column rank, so no column elimination is required before projection.

Table 5: Layer-wise confidence and ranks of highest-confidence incorrect sign predictions produced by *Normal Alignment* and *Future Toggle* [5] on the CIFAR-10 and MNIST models.

Model	Method	Metric	Hidden Layer							
			L1	L2	L3	L4	L5	L6	L7	L8
CIFAR-10 192-64×8-10	<i>Normal Alignment</i>	I-Confidence (%)	✓	50.50	51.00	51.50	55.50	64.00	55.50	55.50
	$n_{\text{attempt}} = 200$	I-Rank		64/64	64/64	61/64	60/64	50/64	60/64	58/64
	<i>Future Toggle</i>	I-Confidence (%)	✓	51.05	53.01	54.55	56.89	55.70	58.72	100.00
	$n_{\text{attempt}} = 200$	I-Rank		61/64	42/64	28/64	15/64	26/64	12/64	1/64
	<i>Future Toggle</i>	I-Confidence (%)	✓	✓	✓	50.62	51.90	53.36	55.58	70.00
	$n_{\text{attempt}} = 1000$	I-Rank				60/64	44/64	45/64	20/64	13/64
MNIST 64-96×3-32-10	<i>Normal Alignment</i>	I-Confidence (%)	55.43	62.30	60.11	56.99	—	—	—	—
	$n_{\text{attempt}} = 200$	I-Rank	88/96	68/96	78/96	29/32	—	—	—	—
	<i>Future Toggle</i>	I-Confidence (%)	50.56	60.71	80.00	100.00	—	—	—	—
	$n_{\text{attempt}} = 200$	I-Rank	96/96	27/96	4/96	1/32	—	—	—	—
	<i>Future Toggle</i>	I-Confidence (%)	✓	57.46	63.77	100.00 [†]	—	—	—	—
	$n_{\text{attempt}} = 1000$	I-Rank		49/96	15/96	1/32	—	—	—	—

I-Confidence: It is the highest confidence among all the incorrect sign predictions. Therefore, the sign-recovery method with lower I-Confidence is better.

I-Rank: Confidence ranks are computed in descending order among all neurons in the corresponding layer; rank 1 denotes the highest confidence. I-Rank is the confidence rank of the highest-confidence incorrect sign prediction. This metric is critical for confidence-ordered enumeration in complete sign recovery: following [13], all signs at or below the I-Rank in the confidence ordering, including the sign at the I-Rank itself, must be included in the enumeration. Therefore, a method is better when its highest-confidence error occurs lower in the confidence ordering, i.e., at a larger numerical I-Rank.

✓: No incorrect sign prediction is produced in this layer.

†: Dual-point utilization in the final hidden layer is only 0.21%; there is a neuron with only two valid votes, both of which are incorrect.

Table 6: Layer-wise comparison of *Normal Alignment*, *eSOE+Alignment*, *Future Toggle*, and *eSOE+Toggle* on a CIFAR-10 model with architecture 192-64×8-10.

Method	Metric	L1	L2	L3	L4	L5	L6	L7	L8	All hidden layers
Normal Alignment $n_{\text{attempt}} = 200$	Correct signs	64/64	63/64	63/64	60/64	62/64	62/64	60/64	61/64	495/512
	Vote accuracy p (%)	75.22	69.83	69.65	68.46	70.73	70.33	70.30	71.05	70.70
	η_{final} (%)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	Time	$2^{1.00}(s)$	$2^{3.32}(s) + 2^1(g)$	$2^{3.17}(s) + 2^{21}(g)$	$2^{3.32}(s) + 2^4(g)$	$2^{3.32}(s) + 2^{21}(g)$	$2^{3.32}(s) + 2^{21}(g)$	$2^{3.32}(s) + 2^{21}(g)$	$2^{3.46}(s) + 2^7(g)$	$2^{6.17}(s) + 2^{15}(g)$
	Queries	$2^{28.11}$	$2^{28.11}$	$2^{28.12}$	$2^{28.13}$	$2^{28.13}$	$2^{28.15}$	$2^{28.15}$	$2^{28.15}$	$2^{31.13}$
	I-Confidence (%)	✓	✓	✓	✓	✓	✓	✓	✓	✓
eSOE+ Alignment $n_{\text{attempt}} = 200$	Correct signs	64/64	64/64	64/64	64/64	64/64	64/64	64/64	64/64	512/512
	Vote accuracy p (%)	75.22	69.83	69.65	68.46	70.73	70.33	70.30	71.05	70.70
	η_{final} (%)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	Time	$2^{6.88}(s)$	$2^{7.04}(s)$	$2^{7.18}(s)$	$2^{7.40}(s)$	$2^{7.82}(s)$	$2^{7.82}(s)$	$2^{9.57}(s)$	$2^{9.43}(s)$	$2^{10.73}(s)$
	Queries	$2^{28.11}$	$2^{28.11}$	$2^{28.12}$	$2^{28.13}$	$2^{28.13}$	$2^{28.15}$	$2^{28.15}$	$2^{28.15}$	$2^{31.13}$
	I-Confidence (%)	✓	✓	✓	✓	✓	✓	✓	✓	✓
Future Toggle $n_{\text{attempt}} = 200$	Correct signs	64/64	64/64	64/64	64/64	64/64	64/64	64/64	64/64	425/512
	Vote accuracy p (%)	100.00	99.78	94.86	91.73	87.38	80.84	71.90	53.77	62.29
	η_{final} (%)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	Time	$2^{7.97}(s) + 2^4(g)$	$2^{6.48}(s) + 2^{24}(g)$	$2^{6.06}(s) + 2^{23}(g)$	$2^{7.21}(s) + 2^{37}(g)$	$2^{7.15}(s) + 2^{39}(g)$	$2^{8.26}(s) + 2^{35}(g)$	$2^{8.26}(s) + 2^{35}(g)$	$2^{9.64}(g)$	$2^{10.85}(s) + 2^{64}(g)$
	Queries	$2^{29.12}$	$2^{29.33}$	$2^{29.44}$	$2^{29.60}$	$2^{29.81}$	$2^{30.08}$	$2^{30.59}$	$2^{31.57}$	$2^{33.17}$
	I-Confidence (%)	✓	✓	✓	✓	✓	✓	✓	✓	✓
eSOE+ Toggle $n_{\text{attempt}} = 200$	Correct signs	64/64	64/64	64/64	51/64	47/64	49/64	50/64	36/64	425/512
	Vote accuracy p (%)	100.00	99.78	94.86	91.73	87.38	80.84	71.90	53.77	62.29
	η_{final} (%)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	Time	$2^{8.55}(s)$	$2^{7.77}(s)$	$2^{7.93}(s)$	$2^{8.15}(s) + 2^{37}(g)$	$2^{8.31}(s) + 2^{39}(g)$	$2^{8.66}(s) + 2^{39}(g)$	$2^{9.26}(s) + 2^{35}(g)$	$2^{9.56}(s) + 2^{64}(g)$	$2^{11.65}(s) + 2^{64}(g)$
	Queries	$2^{29.12}$	$2^{29.33}$	$2^{29.44}$	$2^{29.60}$	$2^{29.81}$	$2^{30.08}$	$2^{30.59}$	$2^{31.57}$	$2^{33.17}$
	I-Confidence (%)	✓	✓	✓	✓	✓	✓	✓	✓	✓
Future Toggle $n_{\text{attempt}} = 1000$	Correct signs	64/64	64/64	64/64	61/64	59/64	57/64	54/64	47/64	470/512
	Vote accuracy p (%)	100.00	99.78	94.86	91.73	87.38	80.84	71.90	53.77	62.29
	η_{final} (%)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	Time	$2^{10.26}(s)$	$2^{9.44}(s)$	$2^{9.36}(s)$	$2^{9.24}(s) + 2^5(g)$	$2^{9.24}(s) + 2^{21}(g)$	$2^{9.89}(s) + 2^{20}(g)$	$2^{10.51}(s) + 2^{15}(g)$	$2^{11.66}(s) + 2^{52}(g)$	$2^{13.17}(s) + 2^{52}(g)$
	Queries	$2^{31.44}$	$2^{31.64}$	$2^{31.76}$	$2^{31.92}$	$2^{32.13}$	$2^{32.41}$	$2^{32.92}$	$2^{33.87}$	$2^{35.49}$
	I-Confidence (%)	✓	✓	✓	✓	✓	✓	✓	✓	✓
eSOE+ Toggle $n_{\text{attempt}} = 1000$	Correct signs	64/64	64/64	64/64	64/64	64/64	64/64	55/64	40/64	469/512
	Vote accuracy p (%)	100.00	99.78	94.86	91.73	87.38	80.84	71.90	53.77	62.29
	η_{final} (%)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	Time	$2^{10.41}(s)$	$2^{9.44}(s)$	$2^{9.36}(s)$	$2^{9.24}(s) + 2^5(g)$	$2^{9.24}(s) + 2^{21}(g)$	$2^{9.89}(s) + 2^{20}(g)$	$2^{10.51}(s) + 2^{15}(g)$	$2^{11.66}(s) + 2^{52}(g)$	$2^{13.17}(s) + 2^{52}(g)$
	Queries	$2^{31.44}$	$2^{31.64}$	$2^{31.76}$	$2^{31.92}$	$2^{32.13}$	$2^{32.41}$	$2^{32.92}$	$2^{33.87}$	$2^{35.49}$
	I-Confidence (%)	✓	✓	✓	✓	✓	✓	✓	✓	✓
I-Confidence/I-Rank:	Correct signs	64/64	64/64	64/64	64/64	64/64	64/64	64/64	64/64	469/512
	Vote accuracy p (%)	100.00	99.78	94.86	91.73	87.38	80.84	71.90	53.77	62.29
	η_{final} (%)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	Time	$2^{10.26}(s)$	$2^{9.44}(s)$	$2^{9.36}(s)$	$2^{9.24}(s) + 2^5(g)$	$2^{9.24}(s) + 2^{21}(g)$	$2^{9.89}(s) + 2^{20}(g)$	$2^{10.51}(s) + 2^{15}(g)$	$2^{11.66}(s) + 2^{52}(g)$	$2^{13.17}(s) + 2^{52}(g)$
	Queries	$2^{31.44}$	$2^{31.64}$	$2^{31.76}$	$2^{31.92}$	$2^{32.13}$	$2^{32.41}$	$2^{32.92}$	$2^{33.87}$	$2^{35.49}$
	I-Confidence (%)	✓	✓	✓	✓	✓	✓	✓	✓	✓
Min confidence:	Correct signs	64/64	64/64	64/64	64/64	64/64	64/64	64/64	64/64	469/512
	Vote accuracy p (%)	100.00	99.78	94.86	91.73	87.38	80.84	71.90	53.77	62.29
	η_{final} (%)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	Time	$2^{10.26}(s)$	$2^{9.44}(s)$	$2^{9.36}(s)$	$2^{9.24}(s) + 2^5(g)$	$2^{9.24}(s) + 2^{21}(g)$	$2^{9.89}(s) + 2^{20}(g)$	$2^{10.51}(s) + 2^{15}(g)$	$2^{11.66}(s) + 2^{52}(g)$	$2^{13.17}(s) + 2^{52}(g)$
	Queries	$2^{31.44}$	$2^{31.64}$	$2^{31.76}$	$2^{31.92}$	$2^{32.13}$	$2^{32.41}$	$2^{32.92}$	$2^{33.87}$	$2^{35.49}$
	I-Confidence (%)	✓	✓	✓	✓	✓	✓	✓	✓	✓
Time:	Correct signs	64/64	64/64	64/64	64/64	64/64	64/64	64/64	64/64	469/512
	Vote accuracy p (%)	100.00	99.78	94.86	91.73	87.38	80.84	71.90	53.77	62.29
	η_{final} (%)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	Time	$2^{10.26}(s)$	$2^{9.44}(s)$	$2^{9.36}(s)$	$2^{9.24}(s) + 2^5(g)$	$2^{9.24}(s) + 2^{21}(g)$	$2^{9.89}(s) + 2^{20}(g)$	$2^{10.51}(s) + 2^{15}(g)$	$2^{11.66}(s) + 2^{52}(g)$	$2^{13.17}(s) + 2^{52}(g)$
	Queries	$2^{31.44}$	$2^{31.64}$	$2^{31.76}$	$2^{31.92}$	$2^{32.13}$	$2^{32.41}$	$2^{32.92}$	$2^{33.87}$	$2^{35.49}$
	I-Confidence (%)	✓	✓	✓	✓	✓	✓	✓	✓	✓
I-Confidence/I-Rank:	Correct signs	64/64	64/64	64/64	64/64	64/64	64/64	64/64	64/64	469/512
	Vote accuracy p (%)	100.00	99.78	94.86	91.73	87.38	80.84	71.90	53.77	62.29
	η_{final} (%)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	Time	$2^{10.26}(s)$	$2^{9.44}(s)$	$2^{9.36}(s)$	$2^{9.24}(s) + 2^5(g)$	$2^{9.24}(s) + 2^{21}(g)$	$2^{9.89}(s) + 2^{20}(g)$	$2^{10.51}(s) + 2^{15}(g)$	$2^{11.66}(s) + 2^{52}(g)$	$2^{13.17}(s) + 2^{52}(g)$
	Queries	$2^{31.44}$	$2^{31.64}$	$2^{31.76}$	$2^{31.92}$	$2^{32.13}$	$2^{32.41}$	$2^{32.92}$	$2^{33.87}$	$2^{35.49}$
	I-Confidence (%)	✓	✓	✓	✓	✓	✓	✓	✓	✓
Min confidence:	Correct signs	64/64	64/64	64/64	64/64	64/64	64/64	64/64	64/64	469/512
	Vote accuracy p (%)	100.00	99.78	94.86	91.73	87.38	80.84	71.90	53.77	62.29
	η_{final} (%)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	Time	$2^{10.26}(s)$	$2^{9.44}(s)$	$2^{9.36}(s)$	$2^{9.24}(s) + 2^5(g)$	$2^{9.24}(s) + 2^{21}(g)$	$2^{9.89}(s) + 2^{20}(g)$	$2^{10.51}(s) + 2^{15}(g)$	$2^{11.66}(s) + 2^{52}(g)$	$2^{13.17}(s) + 2^{52}(g)$
	Queries	$2^{31.44}$	$2^{31.64}$	$2^{31.76}$	$2^{31.92}$	$2^{32.13}$	$2^{32.41}$	$2^{32.92}$	$2^{33.87}$	$2^{35.49}$
	I-Confidence (%)	✓	✓	✓	✓	✓	✓	✓	✓	✓
Time:	Correct signs	64/64	64/64	64/64	64/64	64/64	64/64	64/64	64/64	469/512
	Vote accuracy p (%)	100.00	99.78	94.86	91.73	87.38	80.84	71.90	53.77	62.29
	η_{final} (%)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
	Time	$2^{10.26}(s)$	$2^{9.44}(s)$	$2^{9.36}(s)$	$2^{9.24}(s) + 2^5(g)$	$2^{9.24}(s) + 2^{21}(g)$	$2^{9.89}(s) + 2^{20}(g)$	$2^{10.51}(s) + 2^{15}(g)$	$2^{11.66}(s) + 2^{52}(g)$	$2^{13.17}(s) + 2^{52}(g)$
	Queries	$2^{31.44}$	$2^{31.64}$	$2^{31.76}$	$2^{31.92}$	$2^{32.13}$	$2^{32.41}$	$2^{32.92}$	$2^{33.87}$	$2^{35.49}$
	I-Confidence (%)	✓	✓	✓	✓	✓	✓	✓	✓	✓

I-Confidence/I-Rank: These metrics are defined in Table 5. A ✓ indicates that no incorrect sign prediction is produced in the corresponding layer. If a method combined with *eSOE* does not recover all signs in a layer, we report the I-Confidence and I-Rank of its underlying statistical method, *Normal Alignment* or *Future Toggle*, because the fallback exhaustive search uses that method's confidence ordering.

Min confidence: The minimum confidence used by *eSOE* to eliminate inactive-neuron columns. N/A indicates that the unprojected stacked coefficient matrix already has full column rank, so no column elimination is required before projection.

Time: $2^x(s)$ denotes the method runtime in seconds, whereas $2^y(g)$ denotes the estimated number of candidate sign assignments required by confidence-ordered exhaustive search for complete recovery. The guessing term is omitted when all signs are correctly recovered.

Table 7: Layer-wise comparison of *Normal Alignment*, *eSOE+Alignment*, *Future Toggle*, and *eSOE+Toggle* on an MNIST model with architecture 64-96×3-32-10.

Method	Metric	L1	L2	L3	L4	All hidden layers
Normal Alignment $n_{\text{attempt}} = 200$	Correct signs	91/96	93/96	93/96	30/32	307/320
	Vote accuracy p (%)	70.95	64.98	69.10	72.33	68.74
	η_{dual} (%)	88.24	89.71	91.05	90.53	89.75
	Time	$2^{8.94}(s) + 2^9(g)$	$2^{9.03}(s) + 2^{29}(g)$	$2^{8.84}(s) + 2^{19}(g)$	$2^{7.25}(s) + 2^4(g)$	$2^{10.67}(s) + 2^{2.9}(g)$
	Queries	$2^{26.96}$	$2^{26.97}$	$2^{26.98}$	$2^{25.40}$	$2^{28.71}$
	I-Confidence (%)	55.43	62.30	60.11	56.99	—
eSOE+ Alignment $n_{\text{attempt}} = 200$	Correct signs	96/96	96/96	96/96	32/32	320/320
	Vote accuracy p (%)	70.95	64.98	69.10	72.33	68.74
	η_{dual} (%)	88.24	89.71	91.05	90.53	89.75
	Time	$2^{9.16}(s)$	$2^{9.29}(s)$	$2^{9.10}(s)$	$2^{9.10}(s)$	$2^{11.29}(s)$
	Queries	$2^{26.96}$	$2^{26.98}$	$2^{26.98}$	$2^{25.40}$	$2^{28.71}$
	I-Confidence (%)	✓	✓	✓	✓	—
Future Toggle $n_{\text{attempt}} = 200$	Correct signs	95/96	86/96	66/96	8/32	255/320
	Vote accuracy p (%)	63.67	57.37	56.85	75.00	60.89
	η_{dual} (%)	92.32	60.42	11.26	0.25	49.23
	Time	$2^{13.30}(s) + 2^1(g)$	$2^{13.35}(s) + 2^{70}(g)$	$2^{13.41}(s) + 2^{93}(g)$	$2^{11.93}(s) + 2^{82}(g)$	$2^{15.11}(s) + 2^{93}(g)$
	Queries	$2^{26.63}$	$2^{26.68}$	$2^{26.77}$	$2^{25.31}$	$2^{28.45}$
	I-Confidence (%)	50.56	60.71	80.00	100.00	—
eSOE+ Toggle $n_{\text{attempt}} = 200$	Correct signs	96/96	83/96	67/96	32/32	278/320
	Vote accuracy p (%)	63.67	57.37	56.85	75.00	60.89
	η_{dual} (%)	92.32	60.42	11.26	0.25	49.23
	Time	$2^{13.31}(s)$	$2^{13.36}(s) + 2^{70}(g)$	$2^{13.41}(s) + 2^{93}(g)$	$2^{12.08}(s)$	$2^{15.13}(s) + 2^{93}(g)$
	Queries	$2^{26.63}$	$2^{26.69}$	$2^{26.77}$	$2^{25.31}$	$2^{28.45}$
	I-Confidence (%)	✓	60.71	80.00	✓	—
Future Toggle $n_{\text{attempt}} = 1000$	Correct signs	96/96	82/96	80/96	19/32	277/320
	Vote accuracy p (%)	63.05	56.72	57.31	71.21	60.32
	η_{dual} (%)	91.93	60.37	11.43	0.21	49.14
	Time	$2^{15.57}(s)$	$2^{15.63}(s) + 2^{48}(g)$	$2^{15.66}(s) + 2^{82}(g)$	$2^{14.21}(s) + 2^{32}(g)$	$2^{17.38}(s) + 2^{82}(g)$
	Queries	$2^{28.95}$	$2^{29.00}$	$2^{29.05}$	$2^{27.63}$	$2^{30.77}$
	I-Confidence (%)	✓	57.46	63.77	100.00	—
eSOE+ Toggle $n_{\text{attempt}} = 1000$	Correct signs	96/96	96/96	72/96	32/32	296/320
	Vote accuracy p (%)	63.05	56.72	57.31	71.21	60.32
	η_{dual} (%)	91.93	60.37	11.43	0.21	49.14
	Time	$2^{15.57}(s)$	$2^{15.64}(s)$	$2^{15.67}(s) + 2^{82}(g)$	$2^{14.25}(s)$	$2^{17.39}(s) + 2^{82}(g)$
	Queries	$2^{28.95}$	$2^{29.01}$	$2^{29.06}$	$2^{27.63}$	$2^{30.77}$
	I-Confidence (%)	✓	✓	63.77	✓	—
I-Confidence/I-Rank:	I-Rank	0.64	0.56	15/96	N/A	0.52
	Min confidence	0.64	0.56	0.52	N/A	0.52

I-Confidence/I-Rank: These metrics are defined in Table 5. A ✓ indicates that no incorrect sign prediction is produced in the corresponding layer. If a method combined with *eSOE* does not recover all signs in a layer, we report the I-Confidence and I-Rank of its underlying statistical method, *Normal Alignment* or *Future Toggle*, because the fallback exhaustive search uses that method's confidence ordering.

Min confidence: The minimum confidence used by *eSOE* to eliminate inactive-neuron columns. N/A indicates that the unprojected stacked coefficient matrix already has full column rank, so no column elimination is required before projection.

Time: $2^x(s)$ denotes the method runtime in seconds, whereas $2^y(g)$ denotes the estimated number of candidate sign assignments required by confidence-ordered exhaustive search for complete recovery. The guessing term is omitted when all signs are correctly recovered.