

# Finite-Precision Error Analysis of Cryptanalytic Model Extraction

Duo Xu<sup>1\*</sup>, Liu Zhang<sup>2\*</sup>, and Zilong Wang<sup>1</sup>

<sup>1</sup> School of Cyber Engineering, Xidian University, Xi'an, China

`xuduo@stu.xidian.edu.cn` `zlwang@xidian.edu.cn`

<sup>2</sup> School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore `liu.zhang@ntu.edu.sg`

**Abstract.** Cryptanalytic model extraction treats the parameters of a neural network as hidden secrets and attempts to recover them from black-box oracle access. Existing attacks have shown that such recovery is possible for several neural-network architectures, but practical evaluations typically assume high-precision target-model evaluation, often in `float64`. This assumption creates a mismatch with deployed machine-learning systems, where models are commonly evaluated in `float32` or lower-precision formats.

We study cryptanalytic signature recovery under finite-precision target-model evaluation. We decompose the recovery pipeline into numerical subproblems, including critical-point localization, finite-difference estimation and perturbed linear-system solving, and analyze how errors are introduced, propagated, and amplified across these stages. We further relate the resulting perturbation amplification to intrinsic properties of the target network, including the geometry of local linear regions and local parameter-to-gradient sensitivity.

In particular, we derive approximate instability boundaries for key precision-sensitive stages of the recovery pipeline. When the target model is evaluated in `float32`, oracle-induced perturbations cannot be removed by higher-precision downstream computation, leading to a substantially higher local error floor than in the `float64` setting. Experimental results suggest that recovery becomes less stable as the network size increases. We also discuss how the finite-precision perspective extends to hard-label recovery and piecewise-affine network components, while smooth non-piecewise-linear activations require additional curvature-aware error modeling.

**Keywords:** Cryptanalytic Model Extraction · Finite-Precision Analysis · Signature Recovery · Numerical Stability · ReLU Neural Networks

## 1 Introduction

Model extraction has recently been studied not only as a black-box imitation problem, but also as a cryptanalytic recovery problem. In this line of work, the

---

\* Duo Xu and Liu Zhang contributed equally to this work.

parameters of a neural network are treated as hidden secrets inside a black-box implementation, and the adversary attempts to recover them through carefully chosen queries. Carlini et al. initiated this direction by showing that the weights and biases of fully connected ReLU networks can be extracted layer by layer from oracle access [3]. Subsequent works improved this framework by developing polynomial-time sign-recovery methods, identifying practical bottlenecks in signature extraction, extending end-to-end extraction to deeper ReLU networks, and considering more restrictive oracle models such as the hard-label setting [11,7,10,4,2,9,1]. Recent work has further extended cryptanalytic extraction beyond fully connected ReLU networks, including PReLU networks with learnable activation slopes [5], recurrent neural networks under both raw-output and hard-label settings [16], and convolutional neural networks with max pooling [6]. These results show that cryptanalytic recovery is not limited to simple fully connected architectures, but can exploit structural properties of richer neural-network models.

Despite this progress, a subtle but important aspect of cryptanalytic model extraction remains underexplored: the numerical precision of the target model. Existing practical evaluations typically assume that the target model is implemented in double-precision floating-point arithmetic, i.e., `float64`. This choice is natural from the attacker’s perspective, since higher model precision helps reduce numerical instability in delicate recovery steps such as critical-point localization, signature extraction, signature merging, and parameter refinement. However, it also creates a mismatch between the precision used to demonstrate extraction attacks and the precision commonly adopted by deployed machine-learning systems. In mainstream machine-learning frameworks such as TensorFlow and PyTorch, `float32` remains the standard default numerical format, while mixed-precision training and inference commonly use lower-precision formats such as `float16` and `bfloat16` to improve speed and reduce memory consumption [15,13,12].

This mismatch raises a natural question that has not been systematically studied: even if a cryptanalytic recovery procedure is empirically effective against `float64` target models, does it remain numerically feasible when the target model is evaluated in the precision commonly used in practice? Answering this question requires more than reporting whether an extraction attack succeeds or fails empirically. Each recovery stage may introduce, propagate, or amplify numerical errors, and without a stage-wise numerical analysis, it is difficult to determine whether a failed extraction is caused by an intrinsic algorithmic limitation or by numerical instability. More importantly, existing studies do not clearly characterize when finite-precision perturbations begin to destabilize key stages of the recovery pipeline. In particular, once the numerical error becomes comparable to the useful local signal, critical-point search may fail to detect valid boundary points, and signature merging may no longer reliably distinguish local signatures corresponding to the same hidden parameter.

**Our contributions.** We study cryptanalytic signature recovery under finite-precision target-model evaluation, with a particular focus on the numerical mech-

anisms that affect practical recovery. Our goal is not only to understand how finite precision degrades the final recovered signature, but also to formulate a question that has not been explicitly isolated in prior work: whether finite precision should be regarded not merely as a source of final numerical inaccuracy, but as a factor that can destabilize the recovery process itself. Our main contributions are as follows.

- **Finite-precision error analysis for cryptanalytic recovery.** We study cryptanalytic model extraction as a finite-precision numerical recovery problem, focusing mainly on raw-output attacks against fully connected ReLU networks. We decompose the recovery pipeline into numerical stages, including critical-point localization, local finite-difference estimation, linear-system solving, merging, and refinement. We analyze how perturbations are introduced, propagated, and amplified across these stages. This identifies the main sources of numerical loss, including critical-point localization error, finite-difference amplification, conditioning-induced error amplification, and layer-wise error propagation. It also explains why recovery error may fluctuate across layers, why recovery often becomes less stable in deeper layers, and why some stages may become unstable even when the corresponding exact-arithmetic attack remains valid. We also discuss the implications of this finite-precision perspective for hard-label recovery and broader piecewise-affine network components.
- **Instability boundaries and the practical limits of target-model extraction.** We identify key precision-sensitive stages in the recovery pipeline, including critical-point search and subsequent signature merging steps. Our analysis characterizes the error conditions under which these key stages become unstable under finite precision. In particular, when the target model is evaluated in `float32`, oracle-induced perturbations are already present in the model outputs and cannot be removed by higher-precision downstream computation. Under such a constrained precision regime, our analysis and experiments suggest that, compared with `float64`, recovery against `float32` target models is more susceptible to finite-precision instability<sup>3</sup>.
- **Target-model-dependent factors underlying recovery instability.** We further relate finite-precision recovery instability to intrinsic properties of the target network. In particular, we identify the geometry of local linear regions and local parameter-to-gradient sensitivity as two complementary model-dependent factors that affect the available identification margin. Our analysis and experiments suggest that these properties vary with network scale and training, leading to different recovery behavior even under the same oracle precision and recovery procedure. This perspective indicates that finite-precision feasibility depends not only on the recovery algorithm, but also on how the target model shapes its local linear-region structure and sensitivity profile.

<sup>3</sup> A repository containing the code and experimental results is available at <https://github.com/CryptAnalystDesigner/Finite-Precision-Feasibility-of-Cryptanalytic-Model-Extraction.git>.

**Outline.** Section 2 presents the preliminaries. Section 3 analyzes finite-precision effects in raw-output signature recovery. Section 4 studies the resulting identification instability and the practical limits under `float32` target-model evaluation. Section 5 extends the discussion to hard-label recovery and broader network components. Section 6 concludes the paper.

## 2 Preliminaries

This section introduces the common mathematical and numerical notions used throughout the paper. We first recall the local affine structure of fully connected ReLU networks, then specify the finite-precision model and the numerical notions needed later, and finally introduce the basic geometric objects used in the raw-output and hard-label settings.

### 2.1 Fully Connected ReLU Networks

**Definition 1 (Fully Connected ReLU Network).** *A fully connected ReLU network  $f_\theta : \mathbb{R}^{d_0} \rightarrow \mathbb{R}^{d_{L+1}}$  with  $L$  hidden layers is defined by*

$$h_0(x) = x, \quad z_\ell(x) = A_\ell h_{\ell-1}(x) + b_\ell, \quad h_\ell(x) = \sigma(z_\ell(x)) \quad (\ell = 1, \dots, L),$$

with output

$$f_\theta(x) = A_{L+1} h_L(x) + b_{L+1},$$

where  $\sigma(t) = \max\{t, 0\}$  is applied coordinate-wise, and

$$\theta = \{(A_\ell, b_\ell)\}_{\ell=1}^{L+1}.$$

A fully connected ReLU network is a continuous piecewise-affine function of its input; a short proof is given in Appendix A.1. The following standard decomposition property will be used repeatedly.

**Proposition 1 (Polyhedral Decomposition [14]).** *Every piecewise affine function  $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$  admits a finite polyhedral subdivision of  $\mathbb{R}^n$  such that, on each cell  $\mathcal{R}$ ,*

$$f(x) = J^\mathcal{R} x + \beta^\mathcal{R}, \quad \forall x \in \mathcal{R}.$$

*These cells are referred to as the linear regions of  $f$ , although the local maps are affine in general.*

Hence, for every layer  $\ell$ , both  $z_\ell$  and  $h_\ell$  are continuous piecewise affine functions of the input. In particular, on each linear region  $\mathcal{R}$ ,

$$h_\ell(x) = J_\ell^\mathcal{R} x + \beta_\ell^\mathcal{R}, \quad \forall x \in \mathcal{R}.$$

## 2.2 Finite-Precision Oracle Perturbation

Throughout the paper, finite-precision perturbations are measured relative to the unit roundoff  $u_{\mathcal{O}}$  of the IEEE 754 floating-point format used by the target oracle. For `float32` and `float64`,

$$u_{\text{float32}} = 2^{-24} \approx 5.96 \times 10^{-8}, \quad u_{\text{float64}} = 2^{-53} \approx 1.11 \times 10^{-16}.$$

Heuristically, the finite-precision oracle output is modeled as

$$\mathcal{O}_{\text{fp}}(x) = \mathcal{O}_{\text{exact}}(x) + e(x), \quad \|e(x)\| \lesssim C(x) u_{\mathcal{O}},$$

up to higher-order terms, where  $C(x)$  captures local numerical amplification. In the black-box setting, the relevant perturbation scale is determined by the precision of the target oracle rather than that used in the adversary’s subsequent computations. In particular, if the target model is evaluated in `float32`, then using `float64` arithmetic in the recovery procedure cannot remove the oracle-side perturbation already present in the returned outputs.

## 2.3 SVD, Conditioning, and Null-Space Stability

Many recovery steps reduce to linear-system solving or null-space estimation. For a matrix  $A \in \mathbb{R}^{m \times n}$  with singular value decomposition  $A = U\Sigma V^\top$ , let

$$\sigma_1(A) \geq \sigma_2(A) \geq \dots \geq \sigma_r(A) = \sigma_{\min}^+(A) > 0, \quad r = \text{rank}(A).$$

We use the effective spectral condition number

$$\kappa_2(A) = \frac{\sigma_1(A)}{\sigma_{\min}^+(A)} = \frac{\sigma_1(A)}{\sigma_r(A)}.$$

Heuristically, for a perturbed system  $(A + e_A)\hat{x} = y + e_y$ , the relative solution error scales with  $\kappa_2(A)$  and the absolute errors are:

$$\frac{\|\hat{x} - x\|}{\|x\|} \lesssim \kappa_2(A) \left( \frac{\|e_A\|}{\|A\|} + \frac{\|e_y\|}{\|y\|} \right), \quad \|\hat{x} - x\| \lesssim \frac{\|e_A\| \|x\| + \|e_y\|}{\sigma_{\min}^+(A)} \quad (1)$$

For rank-deficient matrices, all quantities below are understood with respect to the nonzero singular subspace.

Likewise, in null-space estimation, once the perturbation level reaches the scale of the smallest informative singular directions, SVD-based rank selection and null-space identification become numerically ambiguous.

## 2.4 Attack Setting and Recovery Error

Cryptanalytic model extraction is treated here as a parameter-recovery problem. Given black-box access to a target ReLU network  $f_\theta$ , the adversary seeks to recover

$$\theta = \{(A_\ell, b_\ell)\}_{\ell=1}^{L+1}.$$

Because fully connected ReLU networks admit equivalent reparameterizations, such as neuron permutation and positive rescaling, recovery error is measured only after fixing an equivalent alignment of the target parameters. Accordingly,

$$\widehat{\theta} = \theta^* + e_\theta,$$

where  $\theta^*$  denotes the aligned target parameters and  $e_\theta$  the recovery error. Here, all weight matrices and bias vectors are vectorized and concatenated into  $\theta$ . For neuron-wise recovery, we write

$$\theta_{\ell,\eta} := \begin{bmatrix} a_{\ell,\eta} \\ b_{\ell,\eta} \end{bmatrix},$$

where  $a_{\ell,\eta}$  and  $b_{\ell,\eta}$  denote the incoming weight vector and bias of neuron  $\eta$  in layer  $\ell$ .

**Definition 2 (Recovery Errors).** *After alignment, the absolute and relative recovery errors are defined by*

$$\xi := \|\widehat{\theta} - \theta^*\|_2, \quad \xi_i := |\widehat{\theta}_i - \theta_i^*|,$$

and

$$\text{RelErr}_2(\widehat{\theta}, \theta^*) := \frac{\|\widehat{\theta} - \theta^*\|_2}{\|\theta^*\|_2}, \quad \text{RelErr}_{\text{entry}}(i) := \frac{|\widehat{\theta}_i - \theta_i^*|}{\max\{|\theta_i^*|, \tau\}},$$

where  $\tau > 0$  is a fixed threshold used to avoid numerical instability for near-zero true parameters.

## 2.5 Basic Objects in the Raw-Output Setting

The raw-output recovery pipeline is based on critical points and the local affine changes they reveal.

**Definition 3 (Critical Point and Critical Boundary [3]).** *Let  $z_\eta(x)$  denote the pre-activation value of neuron  $\eta$ . An input  $x$  is called a critical point of  $\eta$  if  $z_\eta(x) = 0$ . The set*

$$\mathcal{W}(\eta) = \{x : z_\eta(x) = 0\}$$

*is called the critical boundary of  $\eta$ . The neuron is active at  $x$  if  $z_\eta(x) > 0$ , and inactive if  $z_\eta(x) < 0$ .*

Critical points mark locations at which the local affine representation changes, and therefore serve as the main geometric objects in raw-output recovery.

## 2.6 Basic Objects in the Hard-Label Setting

The hard-label recovery pipeline is based on geometric information extracted from decision boundaries.

**Definition 4 (Decision Boundary [2]).** For a classifier  $F : \mathbb{R}^n \rightarrow \mathbb{R}^C$ , the decision boundary between classes  $i$  and  $j$  is the set

$$\mathcal{B}_{i,j} = \{x : F_i(x) = F_j(x) \geq F_k(x), \forall k\}.$$

A point  $x \in \mathcal{B}_{i,j}$  is called a *boundary point*. Around a regular boundary point, the decision boundary is locally approximated by a hyperplane, whose normal vector is referred to as the *local boundary normal*.

**Definition 5 (Dual Point and Dual Space [2]).** Let  $\Pi_{\text{left}}$  and  $\Pi_{\text{right}}$  be two locally estimated decision-boundary hyperplanes. A point  $x_{\text{dual}}$  is called a dual point if

$$x_{\text{dual}} \in \Pi_{\text{left}} \cap \Pi_{\text{right}}.$$

The affine subspace passing through  $x_{\text{dual}}$  and parallel to the null space of the corresponding hyperplane system is called the dual space.

A dual point plays a role analogous to that of a ReLU critical point in the hard-label setting, and the associated dual space captures geometric information about an underlying neuron-related boundary structure.

## 3 Error Analysis in the Raw-Output Setting

### 3.1 Motivation for Error Analysis

Studying recovery error is not merely about quantifying final numerical inaccuracy. In cryptanalytic model extraction, the recovery pipeline is sequential and layer-wise, so finite-precision perturbations may corrupt intermediate quantities and cause later stages to become unstable or fail. The following two observations illustrate why exact-arithmetic intuition is insufficient for understanding practical recovery.

*Observation 1: Depth-dependent instability without prefix error.* Current-layer recovery error may exhibit markedly larger fluctuations with depth, even when all preceding layers are fixed to their exact target parameters.

To isolate this effect, we run a controlled experiment in which the recovered parameters of all preceding layers are replaced by the exact target parameters, thereby removing direct prefix-error propagation. The experiment uses the code of Liu et al. [10] on a fully connected MNIST network with input dimension 784 and 8 hidden layers of width 8. The results are shown in Table 1.

As shown in Table 1, the recovery error remains small in the earlier layers but exhibits markedly larger fluctuations in deeper ones. This suggests that deeper-layer recovery can become less stable even after the direct effect of prefix-error propagation is removed.

Table 1: Relative  $L_2$  errors of recovered parameters across layers.

| Layer | Neuron 1               | Neuron 2               | Neuron 3               | Neuron 4               | Neuron 5               | Neuron 6               | Neuron 7               | Neuron 8              |
|-------|------------------------|------------------------|------------------------|------------------------|------------------------|------------------------|------------------------|-----------------------|
| 2     | $2.65 \times 10^{-9}$  | $1.01 \times 10^{-10}$ | $5.79 \times 10^{-11}$ | $6.89 \times 10^{-11}$ | $5.67 \times 10^{-9}$  | $5.26 \times 10^{-11}$ | $2.15 \times 10^{-9}$  | –                     |
| 3     | $4.33 \times 10^{-11}$ | $6.12 \times 10^{-11}$ | $2.65 \times 10^{-9}$  | $1.47 \times 10^{-10}$ | $1.03 \times 10^{-10}$ | $8.46 \times 10^{-11}$ | $1.23 \times 10^{-10}$ | $1.28 \times 10^{-9}$ |
| 4     | $2.24 \times 10^{-7}$  | $5.41 \times 10^{-10}$ | $1.82 \times 10^{-11}$ | $6.04 \times 10^{-10}$ | $1.03 \times 10^{-9}$  | $1.37 \times 10^{-10}$ | –                      | –                     |
| 5     | $2.86 \times 10^{-10}$ | $4.29 \times 10^{-10}$ | $1.77 \times 10^{-8}$  | $3.97 \times 10^{-10}$ | $3.01 \times 10^{-9}$  | –                      | –                      | –                     |

“–” indicates that the corresponding neuron was not recovered under our experimental setting.

*Observation 2: Small step sizes can amplify recovery error.* The attack repeatedly estimates derivative-like quantities from nearby oracle queries. In exact arithmetic, decreasing the perturbation step size can improve the accuracy of local finite-difference approximations. Under finite-precision oracle evaluation, however, numerical errors in nearby outputs are increasingly amplified as the step size decreases, making the resulting local estimates less reliable. As illustrated in Table 2, over the tested range, the recovery error increases as the perturbation step size decreases.

Table 2: Illustrative dependence of current-layer recovery error on the perturbation step size  $\Delta x$ .

| $\ \Delta x\ $             | $28 \times 10^{-6}$   | $28 \times 10^{-5}$    | $28 \times 10^{-4}$    | $28 \times 10^{-3}$    |
|----------------------------|-----------------------|------------------------|------------------------|------------------------|
| Median RelErr <sub>2</sub> | $5.29 \times 10^{-9}$ | $1.87 \times 10^{-10}$ | $1.01 \times 10^{-11}$ | $2.17 \times 10^{-12}$ |

Taken together, these phenomena indicate that finite-precision effects in practical floating-point computation cannot be fully captured by a simple multiplicative error-propagation model. In particular, several observed behaviors are difficult to explain under an exact-arithmetic interpretation alone. This motivates a stage-wise analysis of how perturbations enter the recovery procedure, how they are amplified, and what bounds can be established for the resulting recovery error.

### 3.2 Raw-Output Recovery Pipeline

This subsection follows the standard raw-output extraction pipeline and examines how finite-precision error enters and is amplified at each stage. The purpose is not to restate the full attack procedure, but to use its structure to identify the main numerical error amplification factors.

At a high level, raw-output signature recovery proceeds in a layer-wise manner and consists of the following four stages (the detailed recovery pipeline can be found in [3]):

1. **Critical-point localization.** The attacker locates ReLU critical points, namely points at which the local affine representation changes and therefore

reveals information about hidden neurons. These points provide the local boundary anchors used to construct the subsequent partial-signature recovery equations.

2. **Partial-signature recovery.** Around each recovered critical point, nearby oracle queries are used to construct local linear constraints from finite-difference measurements across adjacent linear regions. Solving the resulting linear system yields a *partial signature*, namely a local quantity informative about the parameters of a hidden neuron, typically up to an unknown scaling factor. The partial nature of this signature arises from the piecewise-affine structure of the prefix network. Around a critical point of a neuron in layer  $\ell$ , the activation pattern of the preceding layer is locally fixed. Neurons that are inactive in layer  $\ell - 1$  therefore suppress the corresponding coordinates in the local recovery equation, so a single critical point generally reveals only a subset of the target signature. Different critical points of the same neuron may induce different prefix activation patterns and hence expose complementary subsets of the same underlying parameter vector. This locality-induced incompleteness motivates the subsequent merging stage.
3. **Partial-signature merging.** Since a single critical point generally provides only a partial view of the target signature, recovery requires combining information from multiple critical points. Partial signatures are compared on their jointly observable coordinates and clustered according to whether they are consistent up to the inherent scaling ambiguity. Signatures attributed to the same hidden neuron are then merged into a common component. As additional critical points reveal complementary coordinates, previously unobserved entries may become visible, allowing the merged component to progressively approach a complete signature.
4. **Precision refinement.** Once a correct cluster has been formed, the associated critical points are reused to construct a more stable global system for that neuron, thereby improving the recovered parameter estimate. The purpose of this stage is to reduce the local recovery error and obtain a more accurate neuron-level signature from the merged critical-point information.

Thus, raw-output signature recovery converts local geometric information revealed by black-box queries into neuron-level parameter estimates. Its practical success depends not only on the existence of the underlying local linear systems but also on whether each stage remains numerically stable.

Fig. 1 summarizes the raw-output recovery pipeline and the main stages at which perturbations are introduced and amplified. In the following subsections, the error analysis is organized according to this four-stage decomposition. Among the four stages, partial-signature recovery carries the most direct local numerical burden, as it combines finite-difference estimation with the solution of a linear system whose conditioning may further amplify perturbations.

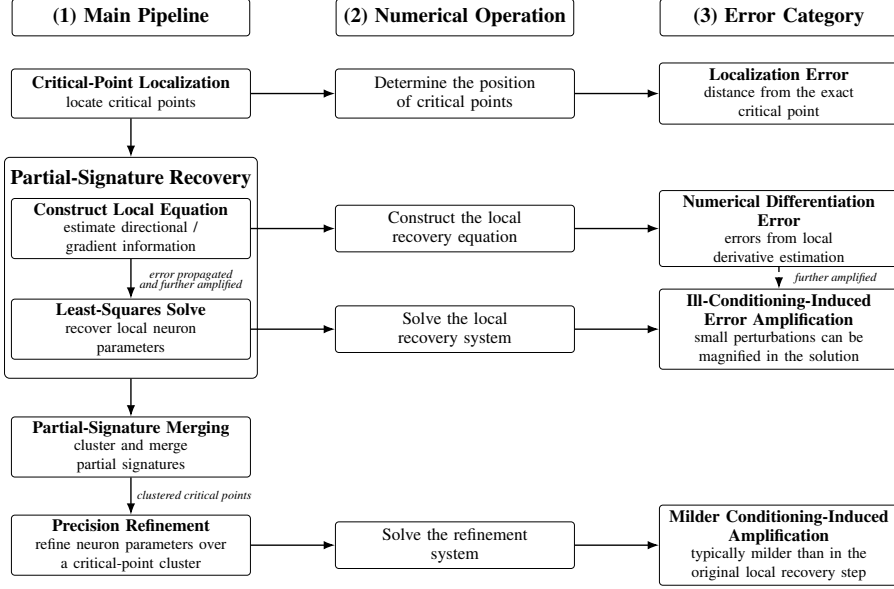


Fig. 1: Raw-output signature recovery pipeline and the main sources of error amplification. The left column shows the recovery stages, the middle column the corresponding numerical operations, and the right column the associated error categories.

### 3.3 Local Error Sources in Signature Recovery

We first analyze the local stages that construct and solve the partial-signature recovery system. These stages are where finite-precision perturbations are first introduced and where the most direct numerical amplification occurs.

*Critical-point Localization* The first step is to localize a critical point from black-box raw-output queries. In practice, this is often done by a line-intersection procedure applied along a search direction [3]. Starting from an interval that is suspected to cross a critical boundary, the attacker compares the directional derivatives at its two sides. If they differ, the two induced local affine lines are intersected to produce a candidate critical point. This candidate is accepted if the discrepancy between the function value predicted by the intersection and the actual oracle output is below a prescribed threshold; otherwise, the interval is bisected and the search continues. If the two directional derivatives coincide, the interval is treated as lying within a single linear region, and the search stops there. The left panel of Fig. 2 illustrates an ideal case in which the search succeeds immediately, the middle panel shows a case that requires another round of binary splitting, and the right panel presents a successful recovery case in the practical setting.

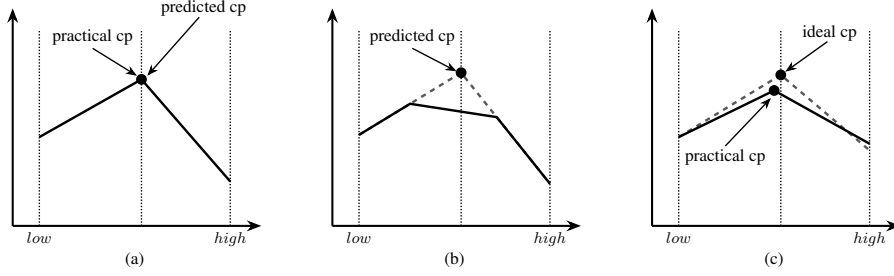


Fig. 2: Illustration of critical-point localization in the raw-output setting. (a) Ideal localization, where the predicted critical point coincides with the practical critical point. (b) A non-ideal case in which the predicted point does not match the practical one. (c) A typical discrepancy between the ideal and practical line segments, where both the slopes and the critical point are slightly perturbed.

The dominant error in this step arises from the perturbation of the intersection point of the two fitted lines. Parameterize the current search segment as

$$x(t) = x_0 + t\Delta, \quad t \in [0, w],$$

where  $\Delta$  is the search direction,  $t$  is the scalar coordinate along this direction, and  $w$  is the width of the current search interval. Let  $t_{\text{cp}}$  and  $\hat{t}_{\text{cp}}$  denote the true and recovered critical-point coordinates, respectively, and define

$$\delta t_{\text{cp}} := \hat{t}_{\text{cp}} - t_{\text{cp}}.$$

Since the parameters of these fitted lines are themselves perturbed, the recovered intersection point is only approximate. Accordingly, the localization error admits the following heuristic estimate:

$$|\delta t_{\text{cp}}| = O\left(\frac{\epsilon_{\text{fit}}}{|\sin \theta|}\right),$$

where  $\epsilon_{\text{fit}}$  denotes the effective perturbation induced by the discrepancy between the ideal local affine lines and the fitted lines, and  $\theta$  is the angle between the two fitted directions. In particular, if the fitting error scales linearly with  $w$ , then

$$\epsilon_{\text{fit}} = O(w).$$

Once the geometric approximation error is no longer dominant and the localization process reaches the oracle precision limit, we heuristically set

$$\epsilon_{\text{fit}} = C(x)u_{\mathcal{O}},$$

where  $u_{\mathcal{O}}$  denotes the oracle-side finite-precision error level. A more explicit derivation is deferred to Appendix A.2.

*Partial-Signature Recovery* The goal of this stage is to recover a local signature of the target neuron from a localized critical point. Let  $\mathcal{R}$  and  $\mathcal{R}'$  be two adjacent linear regions separated by the critical boundary of neuron  $\eta$  in layer  $\ell$ . Under the standard single-switch assumption, the two regions differ only in the activation state of this neuron. Let  $a_{\ell,\eta}$  denote the incoming weight vector of neuron  $\eta$ , and let  $c_\eta^\mathcal{R}$  denote the downstream coefficient that propagates the activation of this neuron to the observed output. Then, in exact arithmetic, the jump of the local gradient across the two regions satisfies

$$\nabla f|_{\mathcal{R}} - \nabla f|_{\mathcal{R}'} = c_\eta^\mathcal{R} (J_{\ell-1}^\mathcal{R})^\top a_{\ell,\eta}, \quad (2)$$

up to the sign convention induced by the ordering of  $\mathcal{R}$  and  $\mathcal{R}'$ . Equation (2) is equivalent to the corresponding gradient-jump identity in [10]; for completeness, a short derivation in our notation is provided in Appendix A.3.

In the black-box setting, this quantity is not accessed analytically, but must be inferred numerically from nearby oracle queries. For a perturbation direction  $\Delta x_i$ , the corresponding jump of the local affine map satisfies

$$\Delta y_i = c_\eta^\mathcal{R} a_{\ell,\eta}^\top J_{\ell-1}^\mathcal{R} \Delta x_i.$$

Sampling multiple perturbation directions

$$\Delta X = [\Delta x_1, \dots, \Delta x_m],$$

gives the linear system

$$M \tilde{a}_{\ell,\eta} = \Delta y, \quad M = (J_{\ell-1}^\mathcal{R} \Delta X)^\top, \quad \tilde{a}_{\ell,\eta} = c_\eta^\mathcal{R} a_{\ell,\eta}, \quad (3)$$

where  $\Delta y$  collects the observed output differences  $\Delta y_i$ . The vector  $\tilde{a}_{\ell,\eta}$  is recovered from this critical point up to a downstream scaling factor and is referred to as a *partial signature*. Equivalently, if  $\Delta X = hV$ , then

$$(J_{\ell-1}^\mathcal{R} \Delta X)^\top = h \tilde{M}, \quad \tilde{M} = (J_{\ell-1}^\mathcal{R} V)^\top.$$

The error in partial-signature recovery has two main sources: numerical error in the construction of  $M$  and  $\Delta y$ , and subsequent amplification during the solution of the linear system.

*Equation-construction error.* The dominant source of error in constructing the system (3) is numerical differentiation, which also provides a natural explanation for the phenomenon reported in Observation 2. In signature recovery, the relevant directional information is estimated through finite differences, whose perturbation step size is denoted by

$$h = \|\Delta x\|_2.$$

The role of this step size can be understood through a classical result from numerical analysis. We recall this result not as a new contribution, but as a standard tool for interpreting the observed finite-precision behavior.

**Proposition 2 (Roundoff Amplification in Finite-Difference Differentiation).** *Let  $f(x) = ax + b$ . Under floating-point evaluations with unit roundoff  $u_{\mathcal{O}}$ , the forward-difference estimator*

$$\widehat{D}_h f(x) = \frac{\text{fl}(f(x+h)) - \text{fl}(f(x))}{h}$$

*satisfies, up to the scale of the queried function values,*

$$\left| \widehat{D}_h f(x) - a \right| \leq \frac{u_{\mathcal{O}}}{h} (|f(x+h)| + |f(x)|).$$

*More generally, if  $F(x) = Ax + c$  on a linear region  $\mathcal{R}$ , and  $v$  is a unit direction such that  $x, x + hv \in \mathcal{R}$ , then the directional finite-difference estimate satisfies*

$$\|\widehat{D}_h F(x; v) - Av\| \leq \frac{u_{\mathcal{O}}}{h} (\|F(x+hv)\| + \|F(x)\|).$$

*again up to the scale of the queried function values.*

The proof of Proposition 2 is deferred to Appendix A.4.

Taking Equation (2) as a heuristic starting point, we have

$$\frac{\nabla f|_{\mathcal{R}} - \nabla f|_{\mathcal{R}'}}{c_{\eta}^{\mathcal{R}}} = (J_{\ell-1}^{\mathcal{R}})^{\top} a_{\ell, \eta}.$$

When the gradients on the two adjacent regions are estimated by finite differences, the perturbation in  $\nabla f|_{\mathcal{R}} - \nabla f|_{\mathcal{R}'}$  is amplified by the normalization factor  $1/c_{\eta}^{\mathcal{R}}$ . Thus, the equation used to construct  $(J_{\ell-1}^{\mathcal{R}})^{\top} a_{\ell, \eta}$  is affected by an error of order

$$O\left(\frac{u_{\mathcal{O}}}{|c_{\eta}^{\mathcal{R}}| h}\right).$$

This formula explains why the error in Observation 2 increases as the step size  $h$  decreases. Empirically, both existing experience and our experiments suggest that this dependence is often close to equality up to constant factors.

*Amplification during linear-system solving.* After the equation construction, we are dealing with a linear system  $M\tilde{a} = \Delta y$  in Equation (3). Using the formulation in Equation (1), the relative perturbations in the linear system are expected to be amplified during the solution process by a factor of order

$$O(\kappa(M)).$$

It remains to understand the conditioning of  $M$ . Since  $M = (J_{\ell-1}^{\mathcal{R}} \Delta X)^{\top}$ , its conditioning is largely governed by  $J_{\ell-1}^{\mathcal{R}}$  up to sampling distortion. Intuitively, random direction sampling observes the local Jacobian through a finite set of directions; when these directions are well spread, they preserve the main singular-value profile of the underlying map. Thus small singular directions of  $J_{\ell-1}^{\mathcal{R}}$  remain the primary source of instability in the sampled recovery system. We therefore use  $\kappa(J_{\ell-1}^{\mathcal{R}})$  as a heuristic proxy for  $\kappa(M)$ .

Finally, the local Jacobian is itself a product of preceding weight matrices and activation masks:

$$J_{\ell-1}^{\mathcal{R}} = D_{\ell-1}^{\mathcal{R}} A_{\ell-1} \cdots D_1^{\mathcal{R}} A_1.$$

As the depth increases, the singular values of this product often become more unevenly distributed, leading to a larger effective condition number. This explains why deeper-layer recovery may become unstable even when the preceding layers are fixed exactly: the current local recovery problem can already be ill-conditioned.

*A local perturbation bound.* The preceding heuristic arguments identify the main numerical amplification factors in the recovery procedure: the finite-difference scale  $h$ , the downstream coefficient  $c_\eta^{\mathcal{R}}$ , and the conditioning of the local recovery matrix  $M$ . We now make this dependence explicit through a local perturbation bound for partial-signature recovery.

**Theorem 1 (Local perturbation bound for partial-signature recovery).**  
Let

$$\Delta x_i = h v_i, \quad \|v_i\|_2 = 1, \quad V = [v_1, \dots, v_m], \quad \tilde{M} := (J_{\ell-1}^{\mathcal{R}} V)^\top.$$

Assume that, under the single-switch condition, the exact gradient-jump measurements satisfy

$$\Delta y = h c_\eta^{\mathcal{R}} \tilde{M} a_{\ell, \eta}.$$

For each query direction  $v_i$ , let the two finite differences be evaluated on the adjacent regions  $\mathcal{R}$  and  $\mathcal{R}'$ , and let

$$\mathcal{Q}_i := \left\{ x_i^{\mathcal{R}}, x_i^{\mathcal{R}} + h v_i, x_i^{\mathcal{R}'}, x_i^{\mathcal{R}'} - h v_i \right\}$$

be the corresponding oracle-query points. Suppose that the finite-precision oracle satisfies

$$O_{\text{fp}}(x) = O_{\text{exact}}(x) + e(x), \quad \|e(x)\| \leq C(x) u_{\mathcal{O}}.$$

Define

$$C_i := \sum_{z \in \mathcal{Q}_i} C(z), \quad C_Q := \left( \sum_{i=1}^m C_i^2 \right)^{1/2}.$$

If  $\tilde{M}$  is exact, then the recovered target parameter, after resolving the scaling ambiguity, satisfies

$$\|e_a\|_2 \leq \frac{\mu_{\mathcal{R}} u_{\mathcal{O}} C_Q}{h \sigma_{\min}^+(\tilde{M})}, \quad \mu_{\mathcal{R}} := \frac{1}{|c_\eta^{\mathcal{R}}|} \quad e_a := \hat{a}_{\ell, \eta} - a_{\ell, \eta}.$$

Consequently,

$$\frac{\|e_a\|_2}{\|a_{\ell, \eta}\|_2} \leq S_Q \mu^{\mathcal{R}} \kappa_2(\tilde{M}) \frac{u_{\mathcal{O}}}{h},$$

where

$$S_Q := \frac{C_Q}{\sigma_{\max}(\tilde{M}) \|a_{\ell, \eta}\|_2}.$$

In particular, under the relative-roundoff model for a scalar-output network, where  $C(x) = |f(x)|$ , the aggregate output scale is

$$C_Q = \left[ \sum_{i=1}^m \left( |f(x_i^{\mathcal{R}} + hv_i)| + |f(x_i^{\mathcal{R}})| + |f(x_i^{\mathcal{R}'} - hv_i)| + |f(x_i^{\mathcal{R}'})| \right)^2 \right]^{1/2}.$$

*Proof sketch.* Each directional measurement is formed from four oracle evaluations, giving

$$\|e_{\Delta y}\|_2 \leq u_{\mathcal{O}} C_Q.$$

Since

$$\Delta y = hc_{\eta}^{\mathcal{R}} \tilde{M} a_{\ell, \eta},$$

using Equation (1), the perturbation in the recovered parameter satisfies

$$\|e_a\| \leq \frac{1}{hc_{\eta}^{\mathcal{R}} \sigma_{\min}^+(\tilde{M})} \|e_{\Delta y}\|.$$

Division by  $\|a_{\ell, \eta}\|_2$  yields the stated relative bound. The complete proof is given in Appendix A.5.

*Remark 1.* Theorem 1 shows that the recovery-error bound contains the same numerical factors identified in the preceding analysis. In particular, the factor  $\mu_{\mathcal{R}} u_{\mathcal{O}} / h$  comes from numerical differentiation, while the factor  $\kappa_2(\tilde{M})$  reflects the amplification introduced when solving the local recovery system. The inequality is intended as an upper bound, with moderate constant factors absorbed into the corresponding error terms.

Empirically, we observe that the actual recovery error is positively correlated with the combined scale

$$O\left(\kappa_2(J_{\ell-1}^{\mathcal{R}}) \mu_{\mathcal{R}} \frac{u_{\mathcal{O}}}{h}\right),$$

where  $\kappa_2(J_{\ell-1}^{\mathcal{R}})$  is used as a heuristic proxy for  $\kappa_2(\tilde{M})$ . It is worth noting that Theorem 1 assumes that the preceding layers have been recovered correctly. The end-to-end recovery error in the full pipeline therefore requires a more detailed analysis.

*Remark 2.* In addition to the amplification factors discussed above, the observed recovery error may also include implementation-dependent amplification, such as solver tolerances and alignment effects, as well as scale-dependent quantities such as the output magnitude  $C(x)$  or the aggregate query scale  $C_Q$ . These factors affect the absolute error level, but in controlled comparisons on the same target model and with the same query strategy, we treat them as constant-scale factors.

### 3.4 Merging, Refinement, and Layer-wise Accumulation

The previous subsection analyzed how perturbations enter and are amplified in local partial-signature recovery. We now discuss the subsequent stages, where recovered local signatures are merged and refined into neuron-level parameter estimates.

*Partial-Signature Merging* Once partial signatures have been recovered from multiple critical points, the next step is to cluster those that are believed to correspond to the same hidden neuron. This merging stage is essential because a single critical point typically yields only a local signature, whereas subsequent refinement requires multiple critical points associated with the same neuron. Although merging does not introduce a further amplification of the existing error in the local recovery system, it is highly sensitive to the quality of the recovered partial signatures: once the pre-refinement error becomes comparable to the similarity threshold used for clustering, signatures from the same neuron may fail to merge. We return to merging stability in Section 4.

*Precision Refinement* If a cluster is correct, then every critical point  $x_i$  in the cluster lies on the same hidden critical hyperplane:

$$a_{\ell,\eta}^\top h_{\ell-1}(x_i) + b_{\ell,\eta} = 0.$$

Stacking these constraints gives the homogeneous refinement system

$$\bar{\Gamma}_\eta \theta_{\ell,\eta} = 0, \quad \bar{\Gamma}_\eta = [\Gamma \mathbf{1}_N], \quad \theta_{\ell,\eta} = \begin{bmatrix} a_{\ell,\eta} \\ b_{\ell,\eta} \end{bmatrix}, \quad (4)$$

where the  $i$ -th row of  $\Gamma$  is  $h_{\ell-1}(x_i)^\top$ . Unlike the local signature-recovery system, this refinement system is constructed directly from clustered critical points and is solved as a null-space problem.

Under finite precision, the recovered representations are perturbed, so the homogeneous refinement system (4) becomes

$$[\Gamma + e_\Gamma \quad \mathbf{1}_N] \begin{bmatrix} \hat{a}_{\ell,\eta} \\ \hat{b}_{\ell,\eta} \end{bmatrix} = 0,$$

where the  $i$ -th row of  $e_\Gamma$  is  $e_{h_{\ell-1}(x_i)^\top}$ .

This refinement step differs from the original signature-recovery step in an important way. Since the system is constructed directly from critical-point representations, it does not require gradient estimation or directional differentiation. Therefore, it removes the dominant numerical-differentiation error caused by subtracting nearly equal outputs and dividing by a small step size. However, refinement is conditional: it assumes that the preceding merging stage has already produced a correct cluster. If the cluster is incorrect, the refinement system mixes inconsistent constraints from different neurons and therefore loses its geometric meaning.

Refinement also does not remove localization error or prefix error. If

$$\hat{x}_i = x_i + e_{x_i},$$

then

$$h_{\ell-1}(\hat{x}_i) = h_{\ell-1}(x_i) + J_{\ell-1}^{\mathcal{R}_i} e_{x_i} + \text{higher-order terms},$$

so localization error directly perturbs the coefficient matrix of the null-space problem. Moreover, the effectiveness of refinement depends on the geometric spread of the sample points used to construct  $\Gamma$ . When the clustered critical points are concentrated in a small region or span only a narrow range of representations, the resulting system may still exhibit instability as  $\Gamma$  is poorly conditioned, even though no explicit finite-difference step is involved. This observation also helps explain why refinement does not perform equally well across different recovery pipelines.<sup>4</sup>

*Layer-wise error accumulation.* In the end-to-end recovery setting, each layer is reconstructed using the refined parameters from the preceding layers as its prefix. Therefore, even after the dominant finite-difference error is reduced by refinement, the current recovery system may still be perturbed by errors inherited from the previous layers. We write this schematically as

$$\hat{\Gamma}_\ell = \Gamma_\ell + e_{\Gamma_\ell}, \quad e_{\Gamma_\ell} = e_{\Gamma_\ell^{\text{cp}}} + e_{\Gamma_\ell^{\text{prefix}}},$$

where the two terms correspond to critical-point localization error and prefix-recovery error, respectively. Since later representations are computed from the recovered prefix,  $e_{\Gamma_\ell^{\text{prefix}}}$  can propagate across layers and accumulate with depth. Thus, refinement improves local recovery accuracy but does not fully remove end-to-end error accumulation, which can still degrade deeper-layer recovery.

## 4 Identification Instability under Finite Precision in the Raw-Output Setting

In Section 3, we quantified the dominant error sources. We now ask whether these perturbations merely accumulate gradually, or whether they can already make precision-sensitive intermediate stages unreliable before later error propagation takes place.

Indeed, the feasibility of these recovery stages depends on whether the underlying structures remain distinguishable in the presence of numerical noise. This turns the recovery process into a sequence of identification problems, where the attack must infer whether perturbed observations correspond to the same underlying structure or to genuinely different ones.

---

<sup>4</sup> This may explain why refinement based on critical points from different linear regions [10] can be more stable than refinement based on locally sampled systems [3].

In this section, we focus on two representative identification stages whose correctness depends directly on numerical precision: the binary decision in critical-point search and the clustering step in partial-signature merging. We show that the piecewise-affine structure of the target network can limit the numerical separation between the underlying structures to be identified, thereby creating a practical bottleneck for their reliable distinction under `float32` or lower precision.

#### 4.1 Identification Instability in Critical-Point Search

In critical-point search, the binary-search procedure determines whether the two endpoints of the current interval belong to the same linear region by comparing their estimated gradients. The reliability of this decision therefore depends directly on the numerical stability of the gradient estimates.

Since the gradients are obtained by finite differences, we heuristically model the estimation error along a chosen direction  $v$  by the local scaling

$$e_{\text{dir}}^{\mathcal{R}}(v; h) = O\left(\left|\frac{\partial f^{\mathcal{R}}}{\partial v}\right| \frac{u_{\mathcal{O}}}{h}\right),$$

where  $h$  is the finite-difference step size. This expression is used only as an order-of-magnitude proxy for the local error scale.

Because the procedure is embedded in a binary search, the finite-difference step must remain within the current interval. If  $w$  denotes the current interval width, then  $h < w$ , and therefore

$$O\left(\left|\frac{\partial f^{\mathcal{R}}}{\partial v}\right| \frac{u_{\mathcal{O}}}{h}\right) \gtrsim O\left(\left|\frac{\partial f^{\mathcal{R}}}{\partial v}\right| \frac{u_{\mathcal{O}}}{w}\right).$$

Hence, as the interval shrinks, the gradient-estimation error is inevitably amplified.

To formalize the resulting instability, suppose that we fix a threshold  $\tau$ , and declare two points to lie in the same linear region whenever the difference between their directional partial derivatives along  $v$  is smaller than  $\tau$ . This rule is reliable only when the numerical error remains negligible compared with  $\tau$ . Once

$$O\left(\left|\frac{\partial f^{\mathcal{R}}}{\partial v}\right| \frac{u_{\mathcal{O}}}{w}\right) \approx \tau, \tag{5}$$

the binary decision becomes unstable and the search becomes highly sensitive to perturbations.

#### 4.2 Identification Instability in Signature Merging

A similar trade-off in the choice of the step size  $h$  also arises in partial-signature recovery. On the one hand,  $h$  must be sufficiently small to ensure that all queried points remain within the same local affine region. On the other hand, the numerical differentiation step in Equation (3) introduces an error of order  $O\left(\frac{\mu_{\mathcal{R}} u_{\mathcal{O}}}{h}\right)$ ,

which becomes smaller as  $h$  increases. Thus, as in critical-point search, smaller  $h$  preserves locality, whereas larger  $h$  improves numerical stability.

This differentiation error is then propagated through the subsequent linear-system solving, where it may be further amplified by the conditioning of the system. We summarize the resulting effect at layer  $\ell$  by an effective partial-signature recovery error  $e_\ell$ . After normalization removes the relative scaling ambiguity, signature merging relies on a threshold-based decision rule, and is therefore directly affected by this accumulated error.

For a fixed hidden neuron  $\eta$ , let  $s_\eta$  denote its normalized ideal signature, and let

$$\hat{s}_{\eta,i} = s_\eta + \xi_{\eta,i}, \quad \|\xi_{\eta,i}\| \leq e_\ell,$$

where  $\hat{s}_{\eta,i}$  and  $\hat{s}_{\eta,j}$  are two recovered partial signatures associated with the same neuron. Then

$$\|\hat{s}_{\eta,i} - \hat{s}_{\eta,j}\| \leq \|\xi_{\eta,i}\| + \|\xi_{\eta,j}\| \leq 2e_\ell.$$

Therefore, if the merging threshold  $\tau_\ell$  is not significantly larger than the within-neuron perturbation scale, that is, if

$$2e_\ell \geq \tau_\ell,$$

then partial signatures that should be merged may already appear too far apart. For an order-of-magnitude estimate, this instability regime is reached when

$$O\left(\kappa_2(J) \frac{\mu_{\mathcal{R}} u_{\mathcal{O}}}{h}\right) \approx \tau_\ell.$$

In this regime, the relevant issue is no longer merely the numerical accuracy of individual partial signatures, but their identifiability. The merging stage must determine whether two perturbed signatures originate from the same hidden neuron; once the within-neuron perturbation becomes comparable to the separation between candidate signatures, the two cases can no longer be reliably distinguished by the merging criterion.

Once these two cases are no longer reliably distinguishable, almost no choice of  $\tau_\ell$  can recover a reliable clustering: decreasing the threshold leads to false splits, whereas increasing it leads to false merges. Consequently, the resulting cluster assignment itself becomes unreliable, and since precision refinement presupposes a correctly identified cluster, the refinement procedure can no longer be meaningfully applied.

### 4.3 Intrinsic Network Factors Governing Identification Instability

The previous two subsections show that finite-precision perturbations may already destabilize two core decision steps of the recovery pipeline. Importantly, whether these instability regimes are reached is not determined by oracle precision or implementation alone, but also by the intrinsic properties of the target network. Two factors are particularly relevant: the local geometry, which constrains the admissible finite-difference scale, and the target-model-dependent observability of parameter-induced gradient changes, as reflected by the downstream factor  $\mu_{\mathcal{R}}$  and the local prefix Jacobian  $J_{\ell-1}^{\mathcal{R}}$ .

*Local-region geometry.* The first factor comes from the piecewise-affine geometry of the target network. Gradient estimation must be performed using queries that remain within the intended local affine region, and therefore the finite-difference step size  $h$  cannot be increased freely. If  $D(\mathcal{R})$  denotes the diameter of a local region, then, at the level of an order-of-magnitude bound,  $h \lesssim D(\mathcal{R})$ . Consequently, under the heuristic scaling, the finite-difference error suggests an intrinsic error scale of order

$$e_{\text{diff}} = O\left(\frac{\mu_{\mathcal{R}} u_{\mathcal{O}}}{D(\mathcal{R})}\right).$$

Thus, smaller local regions force the attack to operate with smaller perturbations and consequently leave less numerical margin for the identification decisions described above.

*Parameter-to-gradient sensitivity.* A second target-dependent factor is the extent to which changes in hidden parameters are reflected in locally observable input gradients. In linear region  $\mathcal{R}$ , for neuron  $\eta$  in layer  $\ell$ , we characterize this relation by the local parameter-to-input-gradient sensitivity operator

$$B_{\eta, \mathcal{R}} := \frac{\partial(\nabla_x f_{\mathcal{R}})}{\partial a_{\ell, \eta}}.$$

Within a fixed ReLU region, on the active side of the corresponding critical boundary,

$$B_{\eta, \mathcal{R}} = c_{\eta}^{\mathcal{R}} (J_{\ell-1}^{\mathcal{R}})^{\top}.$$

A detailed discussion of the equivalence of two equations to represent  $B_{\eta, \mathcal{R}}$  is provided in Appendix A.6.

Viewed through the differential definition, this interpretation is particularly direct:  $B_{\eta, \mathcal{R}}$  measures how strongly the local input-gradient geometry responds to the effective contribution of  $a_{\ell, \eta}$  when the activation state of neuron  $\eta$  changes. A larger gain of  $B_{\eta, \mathcal{R}}$  means that the same activation-induced change in the effective parameter contribution produces a larger observable change in the input gradient.

Hence, the singular spectrum of  $B_{\eta, \mathcal{R}}$  characterizes how strongly perturbations of the target signature are expressed in the observable gradient. In particular, a small effective minimum singular value  $\sigma_{\min}^+(B_{\eta, \mathcal{R}})$  implies a weak local identification margin: distinct parameter perturbations may induce only small differences in the observable gradient and can therefore be more easily masked by finite-precision noise.

*Unified interpretation.* Taken together, these two factors determine the observable local signal. For a query direction  $v$  and step size  $h$ , the change in the local directional response induced when neuron  $\eta$  switches its activation state can be written approximately as

$$\Delta_h f = h v^{\top} B_{\eta, \mathcal{R}} a_{\ell, \eta}.$$

Here, the sensitivity operator  $B_{\eta, \mathcal{R}}$  and the admissible scale of  $h$  are both determined by local properties of the target network. The extraction procedure does not create this sensitivity: it merely probes the intrinsic signal through a directional projection specified by  $v$ . Hence, the available identification margin is fundamentally constrained by the target model itself, rather than by the numerical solver used afterward.

#### 4.4 Experimental Evidence: Limits on Recovering Float32 Model

The experiments in this section show that recovery against `float32` target models is subject to substantial practical limitations. In several regimes, the observed recovery behavior becomes unfavorable, with large effective noise and markedly reduced stability. All experiments in this section are conducted on fully connected neural networks trained on the MNIST training set unless specifically mentioned. The experiments directly validating the theoretical analysis are presented in Appendix B.1. Here, we focus primarily on the overall trends and qualitative behavior of the relevant factors. All subsequent experiments report recovery errors or observations before refinement.

*Local linear-region scale and numerical differentiation* We first conduct a simple experiment to obtain an empirical and intuitive view of the size of local linear regions.

We use two simple empirical proxies to assess the effective size of local linear regions, rather than estimating  $D(\mathcal{R})$  by explicitly locating exact region boundaries. The first is the flip frequency of hidden-neuron activation states along random directions, and the second is the probability that a perturbation of fixed scale  $\epsilon$ , applied at a critical point, remains within the same local linear region. We vary the network width and depth to examine the resulting trends, where  $w \times d$  denotes a fully connected ReLU network with  $w$  neurons per hidden layer and  $d$  hidden layers. Unless otherwise stated, all models are trained on the MNIST training set, take 784-dimensional inputs, and produce one-dimensional outputs. The corresponding results are shown in Tables 3 and 4.

Table 3: Empirical probability that a perturbation remains within the same linear region for different perturbation scales  $\epsilon$ .

| Model/ $\epsilon$ | 2.8    | 0.28   | 0.028  | 0.0028 | 0.00028 |
|-------------------|--------|--------|--------|--------|---------|
| $8 \times 8$      | 0.0786 | 0.4602 | 0.8850 | 0.9878 | 0.9974  |
| $16 \times 8$     | 0.0358 | 0.3092 | 0.8204 | 0.9858 | 0.9990  |
| $256 \times 8$    | 0.0232 | 0.0970 | 0.4670 | 0.9298 | 0.9948  |
| $32 \times 20$    | 0.0850 | 0.2596 | 0.6100 | 0.9146 | 0.9892  |

Table 4: Flip counts for different models, measured along random directions with the one-dimensional search parameter restricted to  $[-28, 28]$ .

| Model size  | $8 \times 8$ | $16 \times 8$ | $256 \times 8$ | $32 \times 20$ |
|-------------|--------------|---------------|----------------|----------------|
| Flip counts | 82           | 198           | 570            | 467            |

We conduct the experiments using randomly chosen directions and report one representative run. Our goal is not to obtain an exact quantitative characterization, but rather to provide an order-of-magnitude comparison. The experimental results suggest that larger networks tend to contain more, and effectively smaller, linear regions. Consequently, the perturbation step size must be reduced to remain within a single local linear region.

*Instability of Critical-Point Search under Linear-Region Constraints.* For the  $8 \times 8$  and  $16 \times 8$  networks, we tested critical-point search with different step sizes and found that the procedure remains feasible for  $h$  in the range from  $10^{-2}$  to  $10^{-3}$ . In contrast, for the  $256 \times 8$  network, critical-point search with  $h = 10^{-2}$  is no longer reliable. Reducing the step size to  $5 \times 10^{-3}$  makes the search feasible again.

*Effect of evaluation precision.* We first compare **float64** and **float32** target-model evaluations on a  $16 \times 8$  network. As shown in Fig. 3, the horizontal axis reports the inverse local sensitivity scale, while the vertical axis reports the relative error of the recovered neurons. Under **float64** evaluation, most recovered neurons remain at a relatively low error level, although a small number of unstable cases can still occur, for example when finite-difference evaluations cross different activation regions. In contrast, **float32** evaluation produces a clear upward shift in the recovery error, where recovery errors shift upward at a scale consistent with the difference in precision between **float32** and **float64**.

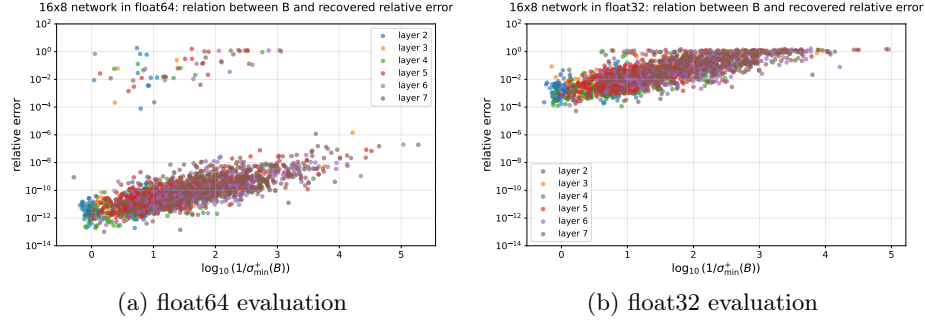


Fig. 3: Float64/float32 comparison for a  $16 \times 8$  network.

The comparison in Fig. 3 is consistent with the error analysis above. Lower evaluation precision increases the perturbation introduced into finite-difference estimates, while the resulting error is further amplified by the local sensitivity and conditioning of the target model. Consequently, `float32` does not merely cause a mild degradation in final recovery accuracy. For sufficiently sensitive neurons, the accumulated error can reach a level that affects the numerical distinguishability required by subsequent partial-signature merging.

*Dependence on target-model properties.* We next examine whether recovery stability can be characterized by network architecture alone. We compare networks with identical architectures but different target-model parameters, including MNIST-trained and randomly initialized networks with widths 8, 16, and 256. All experiments in this comparison use `float64` evaluation.

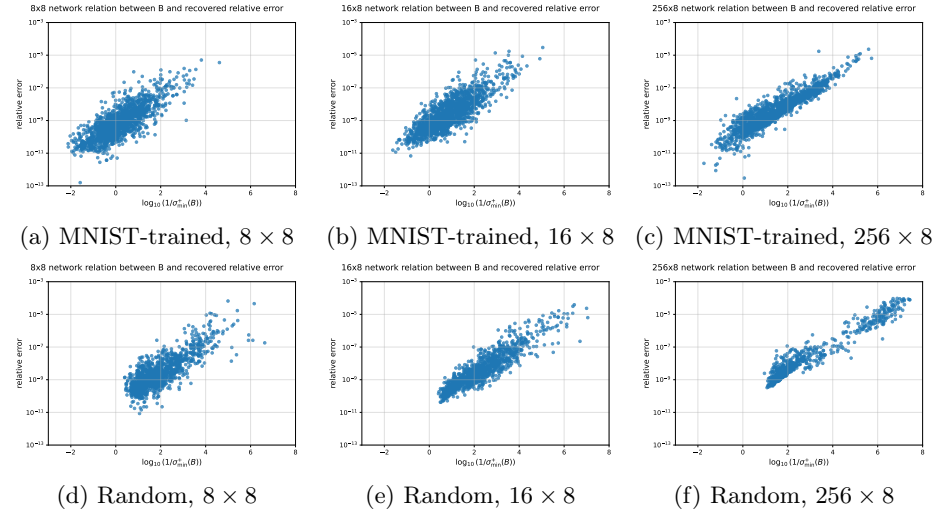


Fig. 4: Relation between the inverse local sensitivity scale and the recovered relative error under `float64` evaluation. The top row shows MNIST-trained target networks, while the bottom row shows randomly initialized target networks. The consistent positive trend supports the interpretation that recovery errors are affected by target-model-dependent local numerical properties, rather than by network width or architecture alone.

As shown in Fig. 4, across both MNIST-trained and random target networks, larger values of the inverse local sensitivity are generally associated with larger recovered errors. This provides further evidence that finite-precision perturbations introduced during the extraction procedure are amplified by target-model-dependent local properties. Moreover, the figure shows that networks with the same architecture can exhibit noticeably different sensitivity distributions and

recovery errors. Hence, recovery stability cannot be characterized by network width or architecture alone; it also depends on the numerical properties of the particular target model.

In fact, the output magnitude and other scale-dependent quantities can also affect the observed recovery error. For example, MNIST-trained models typically produce outputs at the scale of  $10^0$ , whereas the randomly initialized models considered here produce outputs at a larger scale, around  $10^1$ . This difference changes the effective finite-precision perturbation level through the factor  $C(x)$ .

It should be noted that the discrepancy in error here compared to Fig 3 is due to the step size of  $1 \times 10^{-4}$  in this case, whereas the previous one was  $5 \times 10^{-3}$ .

*Summary.* Taken together, the above experiments suggest that, under `float32` or lower oracle precision, the recoverability of larger models becomes questionable. As the network scale grows, more linear regions may become difficult, or even impossible, to identify reliably. This difficulty is caused not only by the reduced geometric size of some regions but also by region-dependent sensitivity.

As a result, the information carried by some activation patterns may become indistinguishable from numerical noise, preventing the reliable recovery of the corresponding partial signatures and potentially compromising the completeness of the recovered network. Understanding how much linear-region information is needed, which types of regions are most informative, and how the training process shapes region-specific properties such as geometric size and parameter-to-gradient sensitivity is important for assessing the feasibility of model recovery. Such questions are more related to neural-network interpretability, and we will leave these questions to future work.

## 5 Beyond Raw-Output Recovery: Discussion on Hard Labels and Broader Network Components

The preceding sections focused on finite-precision error analysis in the raw-output setting, where recovery equations are constructed from numerical outputs and local finite-difference information. We now discuss how this viewpoint changes beyond that setting. In the hard-label setting, the attacker observes only predicted labels, so geometric information must be reconstructed from decision boundaries, introducing additional errors from boundary localization, normal estimation, and dual-space construction. For broader network components, the applicability of the analysis depends on whether the piecewise-affine structure is preserved: the linear-system-based error analysis largely carries over to linear, affine, and piecewise-linear components, whereas smooth or non-piecewise-linear components introduce additional curvature-induced truncation errors.

### 5.1 Finite-Precision Effects and Practical Limits in the Hard-Label Setting

In the hard-label setting, the attacker observes only the predicted label. Thus, the recovery procedure begins by reconstructing local geometric information from

decision boundaries. At a high level, the attack first estimates local decision-boundary hyperplanes, then constructs dual spaces from their intersections, clusters dual spaces associated with the same hidden neuron, and finally solves a linear system. Unlike the raw-output setting, all these geometric objects are estimated from finite label-only queries.

*Recovering local decision-boundary hyperplanes.* The first step is to recover a local hyperplane representation of a decision boundary from black-box label queries. This procedure consists of two substeps. First, a boundary point is localized by binary search between two inputs with different predicted labels. Second, the normal vector of the local boundary hyperplane is estimated from nearby label queries in a way analogous to numerical differentiation.

In the hard-label setting, boundary localization is performed by binary search using only label changes, and thus does not itself require numerical differentiation. Its accuracy is determined by the stopping precision of the search, which can in practice be taken to the scale of  $u_{\mathcal{O}}$ . The main finite-precision amplification arises only in the subsequent finite-difference-type normal-estimation step, yielding an error of order

$$O\left(\frac{u_{\mathcal{O}}}{h}\right).$$

*Dual-space construction.* After obtaining two adjacent decision-boundary hyperplanes and a point  $x_{\text{dual}}$  on their intersection, the attacker constructs the corresponding dual space by solving a hyperplane system. Writing the two local hyperplanes as

$$n_{\text{left}}^{\top}x = n_{\text{left}}^{\top}x_{\text{left}}, \quad n_{\text{right}}^{\top}x = n_{\text{right}}^{\top}x_{\text{right}},$$

we obtain

$$Ax = b, \quad A = \begin{bmatrix} n_{\text{left}}^{\top} \\ n_{\text{right}}^{\top} \end{bmatrix}, \quad b = \begin{bmatrix} n_{\text{left}}^{\top}x_{\text{left}} \\ n_{\text{right}}^{\top}x_{\text{right}} \end{bmatrix}. \quad (6)$$

Here  $n_{\text{left}}$  and  $n_{\text{right}}$  are the recovered normal vectors of the two local decision-boundary hyperplanes, and  $x_{\text{left}}$  and  $x_{\text{right}}$  are the corresponding boundary points. The resulting affine subspace

$$\mathcal{D}_{\text{used}} = x_{\text{dual}} + \text{Null}(A)$$

is referred to as a dual space. A geometric illustration of this construction is given in Fig. 5.

In practice, the estimated quantities are perturbed:

$$\hat{n}_i = n_i + e_{n_i}, \quad \hat{x}_{\text{dual}} = x_{\text{dual}} + e_{\text{dual}}, \quad i \in \{\text{left}, \text{right}\},$$

where  $e_{n_i}$  denotes the normal-estimation error and  $e_{\text{dual}}$  denotes the accumulated localization error of the dual point. Thus, the solver receives

$$\hat{A}x = \hat{b}, \quad \hat{A} = A + e_A, \quad \hat{b} = b + e_b,$$

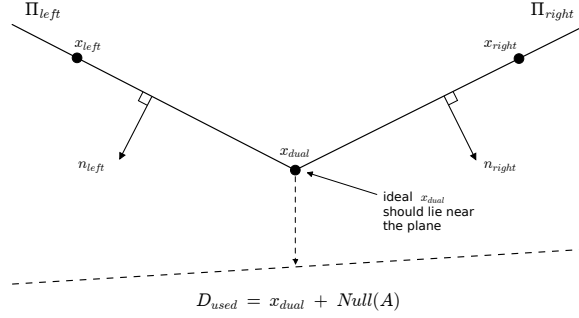


Fig. 5: Geometric interpretation of hard-label signature recovery. The figure illustrates the relationship among the two boundary points  $x_{\text{left}}$ ,  $x_{\text{right}}$ , the dual-space base point  $x_{\text{dual}}$ , and the corresponding boundary normals  $n_{\text{left}}$  and  $n_{\text{right}}$ .

with

$$\Delta A = \begin{bmatrix} e_{n_{\text{left}}}^\top \\ e_{n_{\text{right}}}^\top \end{bmatrix}, \quad e_{b_i} = n_i^\top e_{\text{dual}} + e_{n_i}^\top x_{\text{dual}} + e_{n_i}^\top e_{\text{dual}}.$$

When the perturbations are small, the second-order term can be neglected, giving

$$e_{b_i} \approx n_i^\top e_{\text{dual}} + e_{n_i}^\top x_{\text{dual}}.$$

Thus, the dual-space system contains the same type of numerical differentiation error as in the raw-output setting, because its coefficient matrix is built from estimated normal vectors. In addition, constructing the intersection of two hyperplanes introduces an angle-dependent amplification factor.

**Proposition 3 (Angle-dependent Amplification [8]).** *Let*

$$A = \begin{bmatrix} n_1^\top \\ n_2^\top \end{bmatrix},$$

where  $n_1, n_2$  are unit normal vectors with angle  $\theta$ . Then the singular values of  $A$  are

$$\sigma_{\max}(A) = \sqrt{1 + |\cos \theta|}, \quad \sigma_{\min}(A) = \sqrt{1 - |\cos \theta|}.$$

Hence,

$$\kappa_2(A) = \sqrt{\frac{1 + |\cos \theta|}{1 - |\cos \theta|}}.$$

Therefore, the error in the estimated dual space is amplified when the two decision-boundary hyperplanes have a small angle.

This proposition shows that hard-label recovery has an additional geometric source of instability: even if the two boundary normals are estimated with comparable accuracy, the resulting dual space can be much less accurate when the corresponding hyperplanes are nearly parallel.

By combining the two error sources above, the dual-space perturbation before the final solving stage can be estimated as

$$e_{\text{dual}} = O\left(\sqrt{\frac{1 + |\cos \theta|}{1 - |\cos \theta|}} \left(\frac{u_{\mathcal{O}}}{h} + e_{\text{region}}\right)\right).$$

Here,  $e_{\text{region}}$  denotes the additional error caused by crossing adjacent affine regions. In particular, the perturbation increases when the two boundary hyperplanes become nearly parallel. Detailed experiments on these two amplification effects are provided in Appendix B.2.

*Clustering, SVD-based merging, and final solving.* The recovered dual spaces are clustered and merged before solving the final system (4). This merge is usually an SVD-based consistency test: dual spaces from the same hidden neuron should share a hidden-hyperplane constraint, yielding a matrix with a one-dimensional null space. Finite-precision perturbations from boundary localization, normal estimation, and angle-dependent dual-space construction lift this null direction. When this perturbation reaches the smallest informative singular value, the test becomes numerically ambiguous.

*White-box geometry versus practical black-box recovery.* The above analysis highlights a gap between the idealized geometric description of hard-label recovery and its practical implementation. In the ideal setting, the decision-boundary normals, local boundary hyperplanes, and dual spaces are treated as if they were available with high accuracy. Under such a view, the remaining recovery problem appears mainly linear-algebraic.

In a true hard-label black-box setting, however, these geometric quantities are not directly observable. They must be inferred through finite queries, binary search, projected movement along estimated hyperplanes and numerical normal estimation. This distinction is important because some experimental evaluations of hard-label recovery rely on white-box estimates of gradients and boundary normals, thereby bypassing much of the numerical differentiation error that would appear in a purely query-based setting.

As a result, normal estimation is inherently noisier in the fully black-box regime. We therefore argue that the finite-precision feasibility of hard-label recovery should be evaluated under genuinely query-based normal estimation, rather than through white-box-style approximations.

*Sensitivity interpretation of the boundary angle.* The angle-dependent instability above can also be interpreted through the local parameter-to-gradient sensitivity introduced in Section 4.

Let

$$g_{ij}(x) = F_i(x) - F_j(x)$$

denote the local decision function between classes  $i$  and  $j$ , so that the normal of the corresponding decision boundary in a local affine region  $\mathcal{R}$  is

$$n_{\mathcal{R}} = \nabla_x g_{ij}|_{\mathcal{R}}.$$

Let  $\mathcal{R}'$  denote the adjacent local affine region across the critical boundary of the target neuron  $\eta$ . Under the local single-switch assumption,  $\mathcal{R}$  and  $\mathcal{R}'$  differ only in the activation state of  $\eta$ .

The corresponding parameter-to-gradient sensitivity can be written as

$$B_{\eta}^{(ij)} := \frac{\partial}{\partial a_{\ell, \eta}} (\nabla_x g_{ij}|_{\mathcal{R}'} - \nabla_x g_{ij}|_{\mathcal{R}}) = c_{\eta}^{(ij)} (J_{\ell-1})^{\top}.$$

Accordingly, the change of the decision-boundary normal satisfies

$$n_{diff} := n_{\mathcal{R}'} - n_{\mathcal{R}} = B_{\eta}^{(ij)} a_{\ell, \eta}.$$

Define the relative normal variation

$$\rho := \frac{\|n_{diff}\|}{\|n_{\mathcal{R}}\|},$$

and let  $\phi = \angle(n_{\mathcal{R}}, n_{diff})$ . Then

$$\cos \theta = \frac{1 + \rho \cos \phi}{\sqrt{1 + 2\rho \cos \phi + \rho^2}},$$

where  $\theta$  is the angle between  $n_{\mathcal{R}}$  and  $n_{\mathcal{R}'}$ . Differentiating with respect to  $\rho$  gives

$$\frac{\partial \cos \theta}{\partial \rho} = -\frac{\rho \sin^2 \phi}{(1 + 2\rho \cos \phi + \rho^2)^{3/2}} \leq 0.$$

Thus, for a fixed nonparallel direction of  $n_{diff}$ , the boundary angle increases monotonically with the relative parameter-induced gradient change  $\rho$ . Therefore, the angle governing dual-space conditioning can be viewed as a geometric manifestation of similar target-model-dependent sensitivity, relative to the existing decision-boundary gradient.

*Implications for float32 hard-label recovery.* Under **float32** precision, current hard-label attacks must balance computational cost against numerical accuracy. Since practical fully black-box evaluations at meaningful scale remain limited, and no effective refinement mechanism is currently available to suppress accumulated errors, layer-wise instability may be more severe than in raw-output recovery. This motivates a careful reassessment of the practical feasibility of hard-label extraction under **float32** target-model precision.

## 5.2 Extension to Piecewise-Affine Components

The preceding analysis relies primarily on the local piecewise-affine structure of ReLU networks, rather than on fully connected ReLU networks alone. This observation provides a natural criterion for extending the error analysis: if a network component preserves piecewise-affineness, the recovery problem can still be viewed locally through affine maps, local Jacobians, and linear constraints.

In fact, linear, affine, and piecewise-linear components preserve the local affine geometry of the network. Examples include convolutional layers, fully connected layers, residual additions, average pooling, max pooling, and Batch Normalization in inference mode with fixed statistics. For networks built from such components, the local representation still has the form

$$h_\ell(x) = J_\ell^{\mathcal{R}(x)}x + \beta_\ell^{\mathcal{R}(x)},$$

where  $\mathcal{R}(x)$  denotes the linear region containing  $x$ . Consequently, the same linear-system-based viewpoint remains applicable.

Thus, these components remain compatible with critical-point-based recovery at the structural level. Some architectures may require modified bookkeeping or architecture-specific equation construction, but they do not fundamentally change the finite-precision mechanisms analyzed above. In particular, finite-difference amplification, conditioning-induced error amplification, SVD-based null-space instability, and layer-wise error propagation continue to be the relevant sources of numerical loss.

### 5.3 Non-Piecewise-Linear Components

The preceding analysis relies on a key property of ReLU-based networks: within a fixed linear region, the network behaves exactly as an affine map. Hence, finite-difference-based gradient estimation has no truncation error in exact arithmetic, and the recovery equations can be formulated as exact linear constraints up to finite-precision perturbations. This structure breaks down once a component is no longer piecewise linear. In that case, the network is no longer locally affine on polyhedral regions, and the errors in recovery are no longer limited to roundoff amplification and conditioning-induced amplification.

For smooth non-piecewise-linear components, the main additional effect is curvature-induced truncation error. To illustrate this point, consider the ELU activation

$$\text{ELU}_\alpha(x) = \begin{cases} x, & x \geq 0, \\ \alpha(e^x - 1), & x < 0. \end{cases}$$

On the negative branch,  $\text{ELU}_\alpha$  is smooth but not affine. Thus, for  $x < 0$  and a sufficiently small step size  $h$  such that  $x + h < 0$ , a forward finite-difference approximation contains a deterministic truncation error of order

$$D_h \text{ELU}_\alpha(x) - \text{ELU}'_\alpha(x) = O(h).$$

After including roundoff, the local differentiation error takes the form

$$e_{\text{diff}}(h) \approx C_1 h + C_2 \frac{u}{h},$$

where the first term comes from curvature-induced truncation error and the second from roundoff amplification. Balancing the two terms gives an optimal error scale of order

$$e_{\text{diff}}(h) = \Omega(\sqrt{u}),$$

up to constants depending on the local curvature and the scale of the queried values.

More generally, non-piecewise-linear components break the exact local-affine structure used in ReLU-based recovery. The resulting local constraints are no longer exact linear equations, but approximations whose bias depends on the component’s local nonlinearity. Thus, one must also account for curvature-induced approximation errors and their propagation across layers. Such components do not necessarily make recovery impossible, but they invalidate a direct application of the piecewise-affine error model.

## 6 Conclusion

This paper studies cryptanalytic model extraction as a finite-precision numerical recovery problem rather than an ideal exact-arithmetic procedure. We show that localization error, finite-difference amplification, ill-conditioning and layer-wise error accumulation jointly affect both final parameter accuracy and the stability of intermediate recovery stages. Finite precision is therefore a feasibility issue, not merely an accuracy issue. We further show that this identification margin is target-model-dependent, being shaped jointly by local linear-region geometry and parameter-to-gradient sensitivity. Under realistic `float32` target models, the induced noise floor can become comparable to the useful recovery signal, making the recovery pipeline unstable even before final error dominates. We further discuss how this viewpoint extends to the hard-label setting, piecewise-affine network components, and the limitations posed by smooth activations. These observations motivate future work on noise-robust recovery and feasibility analysis under realistic finite-precision perturbations.

## Acknowledgments

The authors would like to thank anonymous reviewers for their insightful and helpful comments. This research is supported by the National Cryptologic Science Foundation of China (Grant No. 2025NCSF02016), the National Natural Science Foundation of China (Grants No. 62572369).

## References

1. Canales-Martínez, I.A., Santos, D.: Extracting some layers of deep neural networks in the hard-label setting. In: *LATINCRYPT*. pp. 399–421. Lecture Notes in Computer Science, Springer (2025)
2. 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: *EUROCRYPT* (1). pp. 364–396. Lecture Notes in Computer Science, Springer (2025)
3. Carlini, N., Jagielski, M., Mironov, I.: Cryptanalytic extraction of neural network models. In: *CRYPTO* (3). pp. 189–218. Lecture Notes in Computer Science, Springer (2020)

4. Chen, Y., Dong, X., Guo, J., Shen, Y., Wang, A., Wang, X.: Hard-label cryptanalytic extraction of neural network models. In: ASIACRYPT (8). pp. 207–236. Lecture Notes in Computer Science, Springer (2024)
5. Chen, Y., Dong, X., Ma, R., Shen, Y., Wang, A., Yu, H., Wang, X.: Delving into cryptanalytic extraction of prelu neural networks. In: ASIACRYPT (2). pp. 576–607. Lecture Notes in Computer Science, Springer (2025)
6. Chen, Z., Tang, S., Gao, Z., Su, Y., Qin, L., Dong, X.: Algebraic attack on convolutional neural network with max pooling. IACR Cryptol. ePrint Arch. **2026**, 241 (2026)
7. Foerster, H., Mullins, R., Shumailov, I., Hayes, J.: Beyond slow signs in high-fidelity model extraction. In: NeurIPS (2024)
8. Horn, R.A., Johnson, C.R.: Matrix analysis. Cambridge university press (2012)
9. Ito, A., Miura, T., Todo, Y.: Is the hard-label cryptanalytic model extraction really polynomial? IACR Cryptol. ePrint Arch. **2025**, 1868 (2025)
10. Liu, H., Siproudhis, A., Experton, S., Lorenz, P., Boura, C., Peyrin, T.: Navigating the deep: End-to-end extraction on deep neural networks. In: EUROCRYPT. pp. 482–512. Lecture Notes in Computer Science, Springer (2026)
11. Martinez, I.A.C., Chávez-Saab, J., Hambitzer, A., Rodríguez-Henríquez, F., Sathapate, N., Shamir, A.: Polynomial time cryptanalytic extraction of neural network models. In: EUROCRYPT (3). pp. 3–33. Lecture Notes in Computer Science, Springer (2024)
12. PyTorch: Automatic mixed precision package – `torch.amp`. <https://docs.pytorch.org/docs/stable/amp.html>, accessed: 2026-05-21
13. PyTorch: `torch.set_default_dtype`. [https://docs.pytorch.org/docs/stable/generated/torch.set\\_default\\_dtype.html](https://docs.pytorch.org/docs/stable/generated/torch.set_default_dtype.html), accessed: 2026-05-21
14. Scholtes, S.: Introduction to piecewise differentiable equations. Springer (2012)
15. TensorFlow: Mixed precision. [https://www.tensorflow.org/guide/mixed\\_precision](https://www.tensorflow.org/guide/mixed_precision), accessed: 2026-05-21
16. Wei, L., Lei, H., Qi, X., Sun, X., Gao, L., Hu, K., Wang, W., Wang, M.: Cryptanalytic extraction of recurrent neural network models. IACR Cryptol. ePrint Arch. **2026**, 168 (2026)

## A Proofs and Supplementary Analysis

### A.1 Proof of the Piecewise-Affine Property of ReLU Networks

In this appendix, we prove that a fully connected ReLU network defines a continuous piecewise-affine function of its input.

**Lemma 1 (Closure and Coordinate-wise Characterization of Piecewise-Affine Functions [14]).** *Let  $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$  and  $g : \mathbb{R}^m \rightarrow \mathbb{R}^k$  be continuous piecewise-affine functions. Then the set of all possible compositions of the selection functions corresponding to  $f$  and  $g$  forms a collection of selection functions for the composition  $g \circ f$ . Moreover, a continuous function  $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$  is piecewise affine if and only if each of its coordinate functions*

$$f_1, \dots, f_m : \mathbb{R}^n \rightarrow \mathbb{R}$$

*is piecewise affine.*

**Lemma 2 (Closure under Linear Combinations [14]).** *Let  $f_1, \dots, f_r : \mathbb{R}^n \rightarrow \mathbb{R}^m$  be piecewise-affine functions, and let  $\lambda_1, \dots, \lambda_r \in \mathbb{R}$  be fixed scalars. Then the linear combination*

$$F(x) := \sum_{t=1}^r \lambda_t f_t(x)$$

*is piecewise affine.*

**Proposition 4.** *Let*

$$f_\theta : \mathbb{R}^{d_0} \rightarrow \mathbb{R}^{d_{L+1}}$$

*be a fully connected ReLU network with hidden layers*

$$h_0(x) = x, \quad z_\ell(x) = A_\ell h_{\ell-1}(x) + b_\ell, \quad h_\ell(x) = \sigma(z_\ell(x)), \quad \ell = 1, \dots, L,$$

*and output layer*

$$f_\theta(x) = A_{L+1} h_L(x) + b_{L+1},$$

*where  $\sigma(t) = \max\{t, 0\}$  is applied coordinate-wise. Then each  $z_\ell$  and  $h_\ell$  is a continuous piecewise-affine function of  $x$ . In particular,  $f_\theta$  is continuous and piecewise affine.*

*Proof.* We prove the statement by induction on the layer index. The input map  $h_0(x) = x$  is affine, and is therefore continuous and piecewise affine.

Assume that  $h_{\ell-1}$  is continuous and piecewise affine. For each coordinate  $j$ , we have

$$z_{\ell,j}(x) = \sum_{t=1}^{d_{\ell-1}} (A_\ell)_{j,t} h_{\ell-1,t}(x) + (b_\ell)_j.$$

By the coordinate-wise characterization and closure under linear combinations in Lemmas 1 and 2, each coordinate  $z_{\ell,j}$  is continuous and piecewise affine. Hence  $z_\ell$  is continuous and piecewise affine.

Since the scalar ReLU map  $t \mapsto \max\{t, 0\}$  is continuous and piecewise affine, the closure-under-composition property in Lemma 1 implies that each coordinate

$$h_{\ell,j}(x) = \sigma(z_{\ell,j}(x))$$

is also continuous and piecewise affine. Therefore  $h_\ell$  is continuous and piecewise affine.

By induction,  $h_L$  is continuous and piecewise affine. Finally, the output map

$$f_\theta(x) = A_{L+1}h_L(x) + b_{L+1}$$

is an affine combination of the coordinates of  $h_L$ , and is therefore continuous and piecewise affine by Lemma 2.

The same argument applies to other components that preserve piecewise-affine structure under composition and affine combinations. We omit the details.

## A.2 Perturbation of Critical-Point Localization

We consider the one-dimensional restriction of the network to a search line

$$x(t) = x_0 + t\Delta, \quad t \in [0, w].$$

Assume that this interval contains a single critical point  $t_\star$ . Since the network is piecewise affine, the restricted output can be written locally as

$$f(t) = \begin{cases} y_\star + m_1(t - t_\star), & t < t_\star, \\ y_\star + m_2(t - t_\star), & t > t_\star, \end{cases}$$

where  $m_1, m_2 \in \mathbb{R}^q$  are the directional derivatives of the two adjacent affine pieces along the search line. Let

$$g := m_1 - m_2.$$

Using the endpoint values and the derivative  $m_2$ , we obtain

$$f(w) - f(0) - m_2w = gt_\star.$$

Thus, in exact arithmetic, the critical point can be recovered from this linear relation. For vector-valued outputs, we use the least-squares formula

$$t_\star = \frac{g^\top r}{\|g\|^2}, \quad r := f(w) - f(0) - m_2w.$$

Now suppose that the endpoint evaluations and derivative estimates are perturbed:

$$\hat{r} = r + \delta r, \quad \hat{g} = g + \delta g.$$

The recovered critical point is

$$\hat{t}_\star = \frac{\hat{g}^\top \hat{r}}{\|\hat{g}\|^2}.$$

Since  $r = gt_\star$ , we have

$$\hat{t}_\star - t_\star = \frac{\hat{g}^\top \delta r - t_\star \hat{g}^\top \delta g}{\|\hat{g}\|^2}.$$

Therefore, if  $\|\delta g\| \leq \|g\|/2$ , then

$$|\hat{t}_\star - t_\star| \leq \frac{2}{\|g\|} (\|\delta r\| + |t_\star| \|\delta g\|).$$

Since

$$r = f(w) - f(0) - m_2 w,$$

if the endpoint evaluation error is at most  $\eta_y$ , and the derivative estimation error is at most  $\eta_m$ , then

$$\|\delta r\| = O(\eta_y + w\eta_m), \quad \|\delta g\| = O(\eta_m).$$

Consequently,

$$|\hat{t}_\star - t_\star| = O\left(\frac{\eta_y + w\eta_m}{\|m_1 - m_2\|}\right).$$

This estimate shows that localization becomes unstable when the derivative jump  $\|m_1 - m_2\|$  is small. Such cases correspond to flat critical points: the activation pattern changes, but the induced change in the network output is weak. In the scalar-output case, this recovers the usual two-line intersection intuition. If the two graph lines have slopes  $m_1$  and  $m_2$ , then

$$|\sin \theta| = \frac{|m_1 - m_2|}{\sqrt{1 + m_1^2} \sqrt{1 + m_2^2}}.$$

Thus, the geometric factor  $1/|\sin \theta|$  provides an equivalent geometric interpretation of the same instability, while the scale-relevant quantity for the recovery algorithm is the derivative jump  $\|m_1 - m_2\|$ .

### A.3 Derivation of the Local Gradient-Jump Identity

We briefly derive Equation (2) in our notation. The derivation is equivalent in substance to the gradient-jump characterization used in prior work, but we include it here for completeness.

Under the standard single-switch assumption, the activation patterns in the two adjacent regions  $\mathcal{R}$  and  $\mathcal{R}'$  differ only at neuron  $\eta$ , while all other neurons retain the same local affine behavior. On each linear region, the network is affine. Near the boundary, the output can therefore be decomposed as

$$f|_{\mathcal{R}}(x) = c_\eta^{\mathcal{R}} z_{\ell, \eta}(x) + q(x), \quad f|_{\mathcal{R}'}(x) = q(x),$$

up to the choice of sign convention and the ordering of  $\mathcal{R}$  and  $\mathcal{R}'$ . Here,  $q(x)$  collects all terms whose local affine form is unchanged across the boundary, while  $c_\eta^\mathcal{R}$  denotes the downstream coefficient that propagates the contribution of neuron  $\eta$  to the observed output within region  $\mathcal{R}$ .

Taking gradients with respect to the input  $x$ , we obtain

$$\nabla f|_{\mathcal{R}}(x) = c_\eta^\mathcal{R} \nabla z_{\ell,\eta}(x) + \nabla q(x), \quad \nabla f|_{\mathcal{R}'}(x) = \nabla q(x).$$

Subtracting the two identities gives

$$\nabla f|_{\mathcal{R}}(x) - \nabla f|_{\mathcal{R}'}(x) = c_\eta^\mathcal{R} \nabla z_{\ell,\eta}(x).$$

It remains to express  $\nabla z_{\ell,\eta}(x)$  in terms of the incoming weights of neuron  $\eta$ . Since

$$z_{\ell,\eta}(x) = a_{\ell,\eta}^\top h_{\ell-1}(x) + b_{\ell,\eta},$$

the chain rule gives

$$\nabla z_{\ell,\eta}(x) = (J_{\ell-1}^\mathcal{R})^\top a_{\ell,\eta},$$

where  $J_{\ell-1}^\mathcal{R}$  denotes the Jacobian of the prefix map  $h_{\ell-1}(x)$  on region  $\mathcal{R}$ . Substituting this expression into the gradient-jump relation yields

$$\nabla f|_{\mathcal{R}} - \nabla f|_{\mathcal{R}'} = c_\eta^\mathcal{R} (J_{\ell-1}^\mathcal{R})^\top a_{\ell,\eta},$$

which is exactly Equation (2).

#### A.4 Proof of Proposition 2

*Proof.* We first consider the scalar case. Since  $f(x) = ax + b$ , the exact forward difference satisfies

$$\frac{f(x+h) - f(x)}{h} = a.$$

Under the stated floating-point model,

$$\text{fl}(f(x)) = f(x)(1 + \delta_1), \quad \text{fl}(f(x+h)) = f(x+h)(1 + \delta_2),$$

where  $|\delta_1|, |\delta_2| \leq u_{\mathcal{O}}$ . Therefore,

$$\begin{aligned} \widehat{D}_h f(x) - a &= \frac{f(x+h)(1 + \delta_2) - f(x)(1 + \delta_1)}{h} - \frac{f(x+h) - f(x)}{h} \\ &= \frac{\delta_2 f(x+h) - \delta_1 f(x)}{h}. \end{aligned}$$

It follows that

$$\left| \widehat{D}_h f(x) - a \right| \leq \frac{u_{\mathcal{O}}}{h} (|f(x+h)| + |f(x)|),$$

which proves the claimed scalar bound.

The vector-valued locally affine case follows by the same argument. Suppose that  $F(x) = Ax + c$  on a region  $\mathcal{R}$ , and that  $x, x + hv \in \mathcal{R}$ . Then the exact directional finite difference is

$$\frac{F(x + hv) - F(x)}{h} = Av.$$

Under componentwise floating-point perturbations, we write

$$\text{fl}(F(x)) = F(x) + e_1, \quad \text{fl}(F(x + hv)) = F(x + hv) + e_2,$$

where

$$\|e_1\| \leq u_{\mathcal{O}} \|F(x)\|, \quad \|e_2\| \leq u_{\mathcal{O}} \|F(x + hv)\|.$$

Hence

$$\widehat{D}_h F(x; v) - Av = \frac{e_2 - e_1}{h},$$

and consequently

$$\|\widehat{D}_h F(x; v) - Av\| \leq \frac{u_{\mathcal{O}}}{h} (\|F(x + hv)\| + \|F(x)\|).$$

This proves the proposition.

#### A.5 Proof of Theorem 1

*Proof.* Let  $\widehat{\Delta y}$  denote the finite-precision counterpart of  $\Delta y$ . For the  $i$ -th query direction, the corresponding measurement is formed from four oracle evaluations at the points in  $\mathcal{Q}_i$ . Therefore, under the oracle perturbation model,

$$|\widehat{\Delta y}_i - \Delta y_i| \leq u_{\mathcal{O}} \sum_{z \in \mathcal{Q}_i} C(z) = u_{\mathcal{O}} C_i.$$

Stacking the  $m$  measurements gives

$$\widehat{\Delta y} = \Delta y + e_{\Delta y}, \quad \|e_{\Delta y}\|_2 \leq u_{\mathcal{O}} \left( \sum_{i=1}^m C_i^2 \right)^{1/2} = u_{\mathcal{O}} C_Q.$$

By the local recovery equation and the definition of  $\tilde{M}$ , we have

$$\Delta y = c_{\eta}^{\mathcal{R}} h \tilde{M} a_{\ell, \eta},$$

or equivalently,

$$\tilde{M} a_{\ell, \eta} = \frac{\Delta y}{c_{\eta}^{\mathcal{R}} h}.$$

Let

$$\widehat{\Delta y} = \Delta y + e_{\Delta y}$$

be the finite-precision measurement. Assuming that  $c_\eta^{\mathcal{R}}$ ,  $h$ , and  $M$  are exact, the perturbation in the normalized right-hand side is

$$e_{Ma} = e\left(\frac{\Delta y}{c_\eta^{\mathcal{R}} h}\right) = \frac{e_{\Delta y}}{c_\eta^{\mathcal{R}} h}.$$

Therefore, the induced parameter-recovery error satisfies

$$e_a \leq \frac{e_{Ma}}{\sigma_{\min}^+(\tilde{M})} = \frac{e_{\Delta y}}{c_\eta^{\mathcal{R}} h \sigma_{\min}^+(\tilde{M})} = \frac{u_{\mathcal{O}} C_Q}{\sigma_{\min}^+(\tilde{M}) |c_\eta^{\mathcal{R}}| h}.$$

Multiplying both the numerator and the denominator by  $\|M\|_2 = \sigma_{\max}(M)$ , we further obtain

$$\frac{\|e_a\|_2}{\|a_{\ell, \eta}\|_2} \leq \frac{C_Q}{\|\tilde{M}\|_2 \|a_{\ell, \eta}\|_2} \cdot \frac{\|\tilde{M}\|_2}{\sigma_{\min}^+(\tilde{M})} \cdot \frac{u_{\mathcal{O}}}{|c_\eta^{\mathcal{R}}| h}.$$

Using

$$\kappa_2(\tilde{M}) = \frac{\sigma_{\max}(\tilde{M})}{\sigma_{\min}^+(\tilde{M})} = \frac{\|\tilde{M}\|_2}{\sigma_{\min}^+(\tilde{M})}, \quad \mu^{\mathcal{R}} := \frac{1}{|c_\eta^{\mathcal{R}}|},$$

and defining

$$S_Q := \frac{C_Q}{\sigma_{\max}(\tilde{M}) \|a_{\ell, \eta}\|_2},$$

we conclude that

$$\frac{\|e_a\|_2}{\|a_{\ell, \eta}\|_2} \leq S_Q \mu^{\mathcal{R}} \kappa_2(\tilde{M}) \frac{u_{\mathcal{O}}}{h}.$$

This proves the claim.

## A.6 Equivalence of Factorized and Differential Sensitivity Representations

We show that the local sensitivity operator used in Section 4.3 is the same operator that appears in the gradient-jump identity underlying raw-output recovery. We consider a scalar output, or a fixed output coordinate, and adopt the column-gradient convention.

Let  $\mathcal{R}$  be a fixed linear region and denote the Jacobian of the prefix network by

$$J_{\ell-1}^{\mathcal{R}} := \frac{\partial h_{\ell-1}^{\mathcal{R}}(x)}{\partial x} \in \mathbb{R}^{d_{\ell-1} \times d_0}.$$

For neuron  $\eta$  in layer  $\ell$ , define its local parameter-to-gradient sensitivity as

$$B_{\eta, \mathcal{R}} := \frac{\partial(\nabla_x f^{\mathcal{R}}(x))}{\partial a_{\ell, \eta}} \in \mathbb{R}^{d_0 \times d_{\ell-1}}.$$

Because the activation pattern is fixed inside  $\mathcal{R}$ , the network is locally affine. In particular, the contribution associated with neuron  $\eta$  can be separated as

$$f^{\mathcal{R}}(x) = c_{\eta}^{\mathcal{R}} (a_{\ell,\eta}^{\top} h_{\ell-1}^{\mathcal{R}}(x) + b_{\ell,\eta}) + q^{\mathcal{R}}(x),$$

where

$$c_{\eta}^{\mathcal{R}} = \frac{\partial f^{\mathcal{R}}}{\partial z_{\ell,\eta}}$$

is locally constant and  $q^{\mathcal{R}}(x)$  is independent of  $a_{\ell,\eta}$ . Since

$$\nabla_x (a_{\ell,\eta}^{\top} h_{\ell-1}^{\mathcal{R}}(x)) = (J_{\ell-1}^{\mathcal{R}})^{\top} a_{\ell,\eta},$$

we obtain

$$\nabla_x f^{\mathcal{R}}(x) = c_{\eta}^{\mathcal{R}} (J_{\ell-1}^{\mathcal{R}})^{\top} a_{\ell,\eta} + \nabla_x q^{\mathcal{R}}(x).$$

Differentiating with respect to  $a_{\ell,\eta}$  therefore gives

$$B_{\eta,\mathcal{R}} = c_{\eta}^{\mathcal{R}} (J_{\ell-1}^{\mathcal{R}})^{\top}.$$

## B Experiments for Theoretical Validation

To ensure that the validation experiments are numerically feasible, we use models and computations in `float64` precision. All experiments reported below are conducted under this setting.

### B.1 Validation Experiments for the Raw-Output Setting

We first examine the critical-point localization error. According to the geometric estimate in the main text, this error is governed by both the intersection angle  $\theta$  and the search interval width  $w$ . We therefore evaluate the dependence on these two quantities separately, as well as their combined effect. The results are shown in Fig. 6.

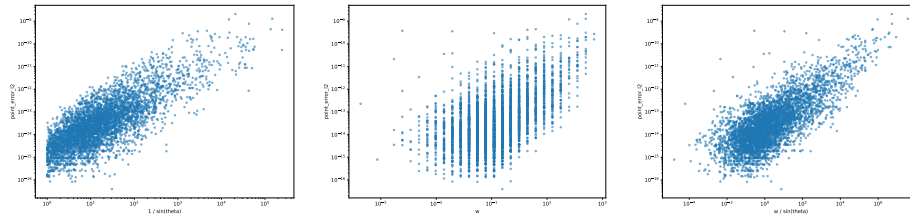


Fig. 6: Empirical correlations for critical-point localization errors. The three plots show the relationship between the localization error and  $1/\sin(\theta)$ ,  $w$ , and  $w/\sin(\theta)$ , respectively. All axes are shown on a logarithmic scale.

Fig. 6 shows that the localization error increases approximately with the geometric amplification factor  $1/\sin\theta$ . When the interval-width error is dominant, the localization error also scales with  $w$ . In contrast, when oracle-side roundoff becomes the dominant perturbation source, the expected error scale is

$$O\left(\frac{u_{\mathcal{O}}}{\sin\theta}\right).$$

These observations are consistent with the estimate that critical-point localization is jointly controlled by angle-dependent amplification and by the dominant perturbation source in the line-fitting step.

*Validation experiment on the relationship between error and finite-difference step size.* We next examine how the finite-difference step size affects the recovery error. According to the error model, decreasing the step size can amplify roundoff errors, whereas increasing it may introduce larger finite-difference approximation errors or increase the risk of crossing into a different local linear region. We therefore evaluate the recovery error over a range of step sizes for two network sizes. The results are shown in Fig. 7 and 8.

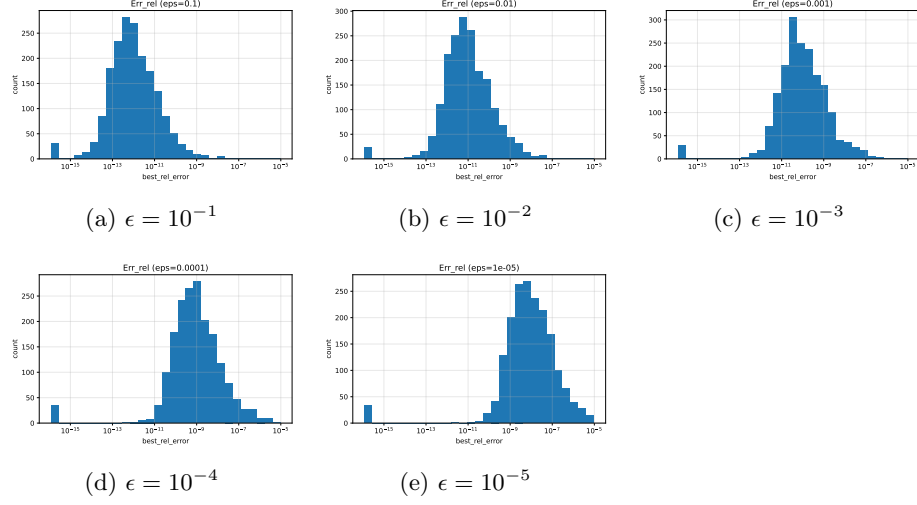


Fig. 7: Relationship between finite-difference step size and relative recovery error for an  $8 \times 8$  network. Each panel shows the histogram of relative errors under a different finite-difference step size  $\epsilon$ .

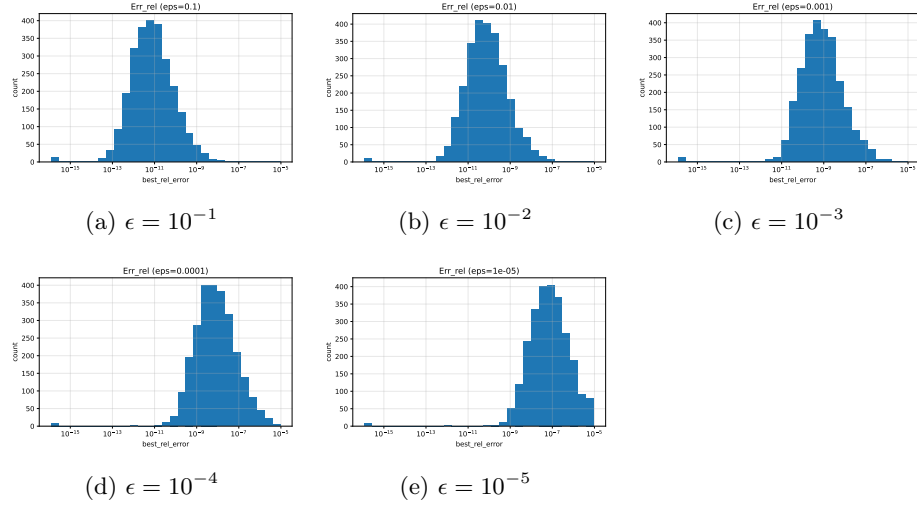


Fig. 8: Relationship between finite-difference step size and relative recovery error for a  $16 \times 8$  network. Each panel shows the histogram of relative errors under a different finite-difference step size  $\epsilon$ .

Figs. 7 and 8 show the relative error distributions for  $8 \times 8$  and  $16 \times 8$  networks under different finite-difference step sizes. In these experiments, the relative error increases as step size  $\epsilon$  becomes smaller. This trend is consistent with the step-size trade-off predicted by the error model: although smaller step sizes can amplify roundoff, overly large step sizes introduce stronger approximation or region-crossing errors.

*Trend of condition number and downstream amplification factor.* We last examine how the recovery error is related to the local condition number, the downstream amplification factor, and their combined effect. The results are shown in Fig. 9.

Fig. 9 compares the relative recovery error with  $\kappa(J)$ , the downstream scaling factor, and their product across different input dimensions. The combined factor provides the clearest explanation of the observed trend, suggesting that the recovery error is jointly governed by the conditioning of the local Jacobian and downstream amplification.

Moreover, these two quantities appear to influence not only the growth trend of the error, but also the empirical error floor. This observation suggests a possible connection between the error floor and the gradient distribution in wider networks, although a precise characterization of this relationship requires further investigation.

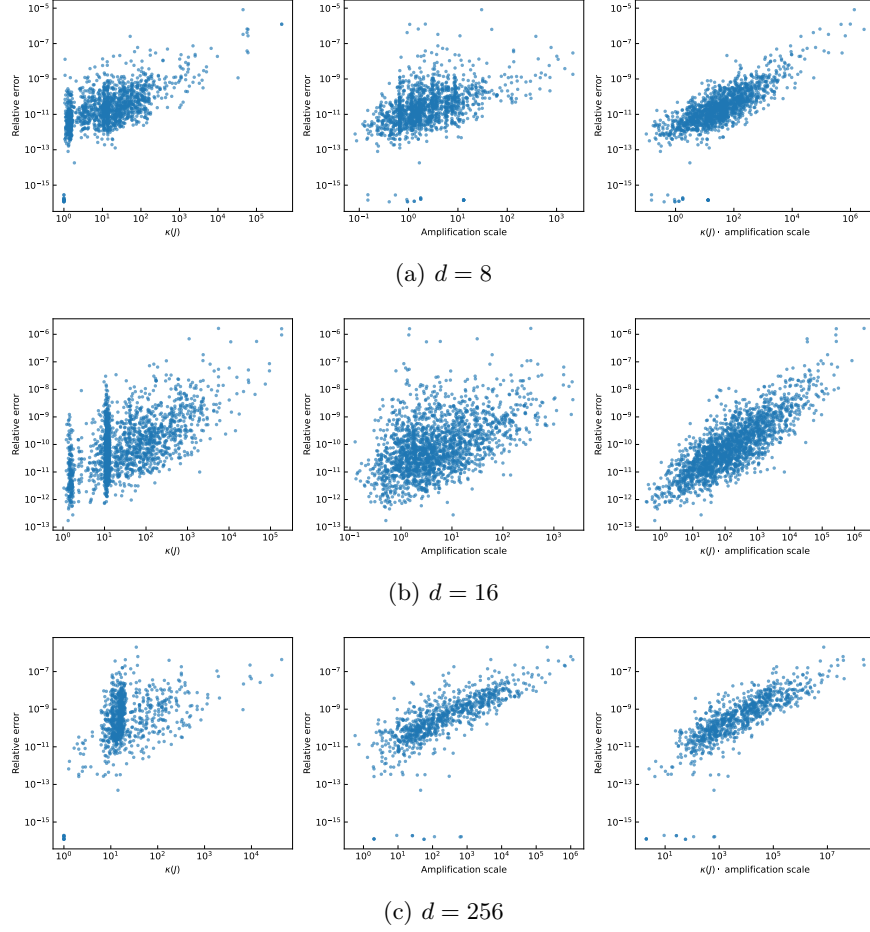


Fig 9: Relationship between relative recovery error, local condition number, and downstream amplification under different input dimensions. All numerical derivatives are computed using the finite-difference step size  $\epsilon = 0.0028$ . In each row, the left panel plots relative error against the condition number  $\kappa(J)$ , the middle panel plots relative error against the downstream scaling factor, and the right panel plots relative error against the combined factor  $\kappa(J) \cdot (\text{downstream scaling factor})$ .

## B.2 Experimental Validation in the Hard-Label Setting

This appendix provides a brief validation of two effects discussed in the main text: the error introduced by numerical differentiation, and its subsequent amplification through the conditioning of the hyperplane-intersection step.

*Decision-boundary normal estimation error.* We first evaluate the recovered decision-boundary normals under two finite-difference step sizes,  $\epsilon = 10^{-6}$  and  $\epsilon = 10^{-3}$ . For each value of  $\epsilon$ , we report the sign-invariant relative  $\ell_2$  errors of the left and right boundary normals in Fig. 10. Each histogram is computed from 3000 sampled normal vectors.

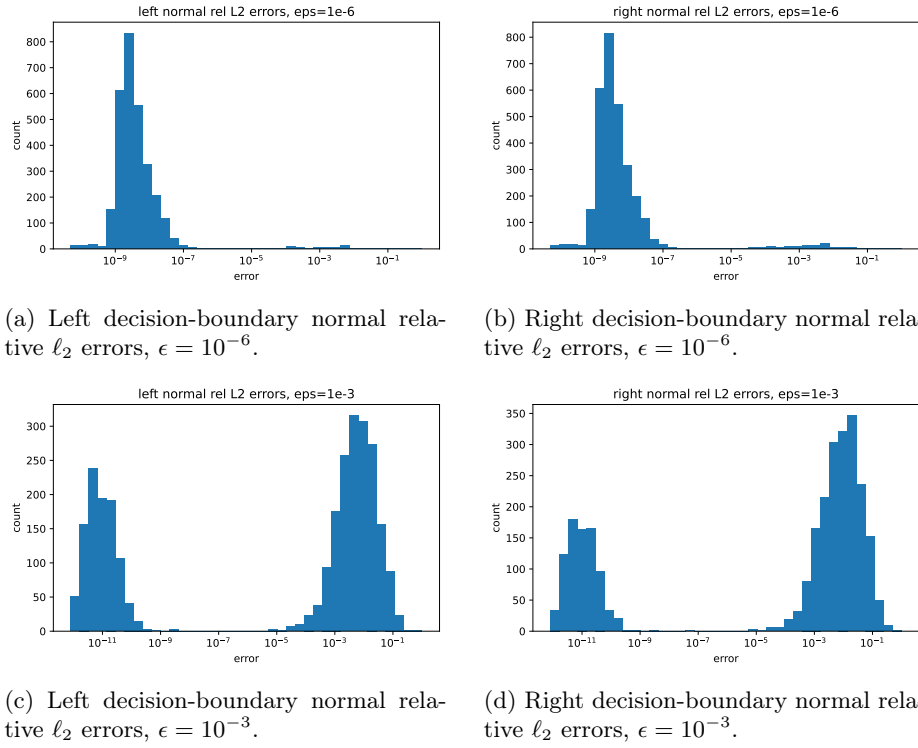


Fig. 10: Histograms of the sign-invariant relative  $\ell_2$  errors of the recovered left and right decision-boundary normals under numerical differentiation with  $\epsilon = 10^{-6}$  and  $\epsilon = 10^{-3}$ . Each histogram is computed from 3000 sampled normal vectors.

As shown in Fig. 10, the choice of finite-difference step size exhibits a clear trade-off. When  $\epsilon = 10^{-6}$ , finite-difference roundoff amplification leads to larger normal-estimation errors, but cross-region events occur less frequently. In contrast, when  $\epsilon = 10^{-3}$ , the numerical differentiation error is smaller, but the larger

step size makes cross-region events more likely. This behavior is consistent with the heuristic error model

$$O\left(\frac{u_{\mathcal{O}}}{\epsilon} + e_{\text{region}}\right),$$

where decreasing  $\epsilon$  amplifies finite-precision error, whereas increasing  $\epsilon$  increases the contribution from region-crossing effects.

*Angle-dependent dual-space amplification.* We next examine the angle-dependent amplification effect in hard-label dual-space construction. In this experiment, we compare the theoretical amplification factor

$$\sqrt{\frac{1 + |\cos \theta|}{1 - |\cos \theta|}}$$

with the empirically observed amplification from normal-estimation errors to the resulting dual-space or triplet errors. The results are shown in Fig. 11.

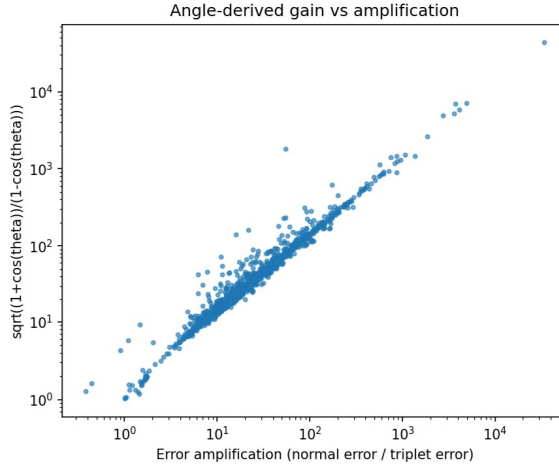


Fig. 11: Relationship between the theoretical angle-dependent amplification factor and the empirically observed error amplification in hard-label dual-space construction. The horizontal axis shows the measured amplification ratio from normal-estimation error to the resulting dual-space or triplet error, while the vertical axis shows the theoretical factor  $\sqrt{(1 + |\cos \theta|)/(1 - |\cos \theta|)}$ . Both axes are shown on a logarithmic scale.

Fig. 11 shows that the observed amplification ratio increases approximately with the theoretical angle-dependent factor. This supports the analysis that dual-space construction becomes increasingly ill-conditioned as the two decision-boundary hyperplanes approach parallelism. In this regime, even modest normal-

estimation errors can be significantly amplified in the subsequent hyperplane-intersection step.