Polynomial-Time Cryptanalytic Extraction of Graph Neural Networks in the Hard-Label Setting

Chun Li[1], Liping Zhuang[1], Di Li[1], Yufeng Tang[2], and Zheng Gong[1*]

[1] School of Computer Science, South China Normal University, Guangzhou, China
springli9807@gmail.com, 2025010261@m.scnu.edu.cn, edu.lidi97@gmail.com, cis.gong@gmail.com

[2] Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing, China
yufengtang@tsinghua.edu.cn

Abstract. Graph neural network parameters are valuable intellectual property, and high-fidelity extraction enables model stealing, evasion, and downstream abuse. Prior cryptanalytic extraction can recover exact parameters in hard-label settings, yet existing methods focus on feedforward models and do not extend to message-passing graph neural networks. The obstacles include coupled queried nodes, hidden aggregation operators, and sign and scale ambiguities created by message passing. To our knowledge, this is the first cryptanalytic extraction framework for message-passing ReLU graph neural networks in hard-label settings where the oracle returns only the predicted class. We locate dual-point constraints on class-pair decision boundaries and activation hyperplanes, then recover hidden signatures with SVD-based extraction under a fixed ordered two-node query interface. We use ON/OFF distance comparisons to resolve hidden signs, then prove that diagonal scale factors propagate through the recovered prefix and cancel at the output layer. This four-stage recovery chain avoids explicit scale fitting while using only hard-label local queries. On ogbn-arxiv targets with up to 10,660 parameters, the attack achieves 100% prediction fidelity on 8,171 evaluated nodes with 48.68M node queries.

Keywords: Graph neural networks · model extraction attacks · hard-label setting · cryptanalytic extraction · black-box security

1 Introduction

Graph Neural Networks support high-value services in fraud detection, recommendation, scientific screening, and security analytics [19,16,29,21]. Their trained parameters therefore encode both economic value and task-specific inductive structure for deployed models [21]. Model extraction attacks therefore raise a parameter-security question for message-passing GNNs under black-box access [28,15,9]. We study that question under a strong hard-label local-query adversary that can repeatedly probe one fixed ordered block and control its node features.

Existing cryptanalytic attacks exploit piecewise-linear geometry and convert boundary events into algebraic constraints on hidden parameters [7,14]. This program works for feedforward architectures because each boundary crossing isolates one local neuron signature. Message passing breaks that separation because the queried node mixes self and neighbor channels under shared weights. One observed bend therefore no longer maps to one independent feedforward equation for a single hidden neuron. The parameter-security question for message-passing ReLU GNNs has therefore remained open under strong hard-label local queries. Prior GNN extraction work therefore remains largely learning-based in this setting across published attacks to date [26]. Our method instead uses a four-stage cryptanalytic recovery chain under a strong local-query adversary. The target can be any queryable GNN API on which the adversary can repeatedly evaluate the same ordered local block and control its features. We analyze the minimal recoverable case of an ordered two-node block, which induces one fixed 2×2 aggregation operator across repeated calls.

Table 1. Comparison of Cryptanalytic Extraction Methods

Type	Activation	Architecture	Params	Hard-label	Signature	Sign	Reference
FCN	ReLU	784-128-1	≈ 0.1M	×	$\mathcal{P}$	$\mathcal{E}$	[18]
FCN	ReLU	10-20-20-1	620	×	$\mathcal{P}$	$\mathcal{E}$	[24]
FCN	ReLU	10-20-20-1	620	×	$\mathcal{P}$	$\mathcal{E}$	[7]
FCN	ReLU	784-128-1	≈ 0.1M	×	$\mathcal{P}$	$\mathcal{E}$	[7]
FCN	ReLU	3072-256×8-10	≈ 1.25M	×	$\mathcal{P}$	$\mathcal{E}$	[4]
FCN	ReLU	100-50-50-1	7,650	×	$\mathcal{P}$	$\mathcal{P}$	[14]
FCN	ReLU	1024-2-2-1	20,508	✓	$\mathcal{P}$	$\mathcal{E}$	[10]
FCN	ReLU	3072-256×3-64-10	~ 0.9M	✓	$\mathcal{P}$	$\mathcal{P}$	[6]
FCN	ReLU	3072-256×3-64-10	~ 0.9M	✓	$\mathcal{P}$	$\mathcal{P}$	[5]
FCN	ReLU	3072-256×8-10	≈ 1.25M	×	$\mathcal{P}$	$\mathcal{P}$	[20]
CNN	ReLU	LeNet-5	61,706	✓	$\mathcal{P}$	$\mathcal{E}$	[27]
CNN	ReLU	LeNet-5-MaxPool	61,706	×	$\mathcal{P}$	$\mathcal{E}$	[12]
RNN	ReLU	800-16×40-1	13,089	×	$\mathcal{P}$	$\mathcal{P}$	[30]
RNN	ReLU	800-16×40-1	13,089	✓	$\mathcal{P}$	$\mathcal{E}$	[30]
FCN	PReLU	784-60-1	47,221	Top m	$\mathcal{P}$	$\mathcal{P}$	[11]
FCN	Sigmoid/Softplus GELU/SiLU/SELU	784-8×8-1	6,280	×	$\mathcal{P}$	$\mathcal{P}$	[2]
FCN	HardTanh	64-64×2-10	8,970	✓	$\mathcal{P}$	$\mathcal{P}$	[23]
FCN	HardTanh	64-64×4-10	17,290	×	$\mathcal{P}$	$\mathcal{P}$	[23]
GNN	**ReLU**	**32-32×5-4**	**10,660**	✓	$\mathcal{P}$	$\mathcal{P}$	**Ours**

Note: Hard-label support is denoted by ✓, unsupported settings by ×, and partial top-label access by Top m. Complexity symbols use $\mathcal{P}$ for polynomial recovery and $\mathcal{E}$ for exponential recovery. For our method, the architecture field lists the target GNN widths, while the attack instantiates only the minimal recoverable two-node block rather than requiring graph-wide node access.

An ordered two-node query block already suffices for exact recovery under the fixed local-query model considered here. Larger query objects can only strengthen the attack because the adversary can reuse the same recoverable sub-block. Table 1 places this reduction within the broader cryptanalytic literature on exact recovery methods. The reduction therefore lets us test whether message-passing GNNs retain parameter security even against a deliberately strong local-query adversary. To our knowledge, no prior cryptanalytic study resolves this parameter-security question for message-passing GNNs under the fixed hard-label local-query model. Each contribution below addresses one obstacle introduced by message passing.

The paper makes the following four contributions under this strong attacker model:

1. It shows that parameter security fails for hard-label message-passing ReLU GNNs under a strong local-query adversary by giving a polynomial-time cryptanalytic extraction attack. It also identifies a minimal recoverable two-node block that is sufficient under the fixed local-query model.
2. It develops a four-stage recovery pipeline that reconstructs hidden and output parameters from hard labels alone.
3. It proves a scale-chain cancellation theorem that removes explicit layer-wise scale fitting during sequential recovery.
4. It validates this insecurity result across datasets and architectures with exact sign recovery, 100% fidelity, and stable numerical error.

Section 3 fixes the core notation used throughout the local-query recovery derivation in the paper. Section 4 develops the four-block recovery chain. Section 5 validates exact recovery on the target GNN and clarifies the resulting defensive risk for local-query APIs. Appendix A collects the executable recovery algorithms used to instantiate the main-text equations in practice. The implementation and evaluation code for the attack pipeline are publicly available in the repository `https://github.com/springli07/GNN_MP_CEA`.

2 Related Work

Learning-Based Model Extraction. Learning-based extraction trains a surrogate model to imitate the victim directly from black-box prediction responses [28,22,13]. Later variants improve sample efficiency through active querying, data-free synthesis, and generative-prior query strategies in black-box settings [9,1,3]. These attacks target behavioral similarity rather than exact parameter recovery, so they do not identify the internal algebraic structure of the target model [21].

Cryptanalytic Extraction. Cryptanalytic extraction leverages piecewise-linear geometry and translates boundary information into algebraic constraints on hidden parameters [8]. Early studies established polynomial-time signature recovery for ReLU networks under increasingly general black-box settings [7,18,4]. Later work improved hard-label sign recovery and strengthened exact extraction guarantees for deeper or wider targets [14,10,6,5]. Recent extensions now cover broader activation families and additional architectures, including CNNs and RNNs explicitly [2,23,27,12,30]. This progression leaves message-passing GNNs as the main unresolved hard-label architecture family in cryptanalytic extraction.

GNN Extraction. GNN extraction studies remain dominated by distillation and surrogate training rather than exact parameter reconstruction [26]. These methods require broad graph-level supervision because they learn behavior over many nodes instead of isolating local neuron equations. To our knowledge, no prior cryptanalytic result proves exact hard-label recovery for message-passing ReLU GNNs under a fixed local-query interface. This paper addresses that gap by identifying a minimal recoverable two-node block for exact recovery.

3 Notions and Notations

The recovery analysis uses a fixed local-query view of a message-passing GNN throughout the paper. We consider an attributed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, X)$. Its node features are $X \in \mathbb{R}^{n \times d_0}$, and its adjacency matrix is $A \in \{0,1\}^{n \times n}$. The normalized aggregator is $\widetilde{A} = (D + I_n)^{-1}(A + I_n)$. The target is an L-layer message-passing GNN with ReLU hidden layers and an identity output layer.

The adversary fixes an ordered local query block $S = (v_1, \ldots, v_m)$ with $m \geq 2$. Its controllable block features are $X_S \in \mathbb{R}^{m \times d_0}$. The same ordered block is reused across repeated calls, so the induced local aggregation operator A_S remains constant throughout recovery. The main text instantiates the minimal case $m = 2$ with $S = (1, 2)$, $X := X_S$, and $A_q := A_S$.

Definition 3.1 (Message-Passing GNN Layer). *Eq. 1 defines the layer map used throughout the recovery derivation for the queried local interface.*

$$\Phi_{\mathrm{GNN}}(H, W_{self}, W_{neigh}, b) = \sigma\left(HW_{self} + \widetilde{A}HW_{neigh} + 1_n b^\top\right). \tag{1}$$

Where $H \in \mathbb{R}^{n \times d_{in}}$ is the layer input, $W_{self}, W_{neigh} \in \mathbb{R}^{d_{in} \times d_{out}}$ are trainable weights, and $b \in \mathbb{R}^{d_{out}}$ is the bias term. This form fixes the self, neighbor, and bias decomposition used in all later recovery equations.

Definition 3.2 (L-Layer GNN). *Eq. 2 defines the full L-layer composition used by the target model throughout the paper.*

$$f(X \mid \Theta) = f^{(L-1)} \circ \cdots \circ f^{(0)}(X). \tag{2}$$

Where $\Theta^{(\ell)} = (W^{(\ell)}_{self}, W^{(\ell)}_{neigh}, b^{(\ell)})$ stores the parameters of layer ℓ in this composition. This indexing convention is used by all later recovery equations in the attack derivation.

Definition 3.3 (Activation Boundary). *Eq. 3 defines the activation boundary of neuron j at queried node v in the local interface.*

$$\mathcal{H}^{(\ell)}_{v,j} = \left\{X_S \in \mathbb{R}^{m \times d_0} : z^{(\ell)}_{v,j}(X_S) = 0\right\}. \tag{3}$$

Where $z^{(\ell)}_{v,j}(X_S)$ is the pre-activation of neuron j at queried node $v \in S$. This boundary supplies the local zero set used in dual-point discovery.

Definition 3.4 (Decision Boundary). *Eq. 4 defines the node-level decision boundary for the queried local interface used in recovery.*

$$\mathcal{B}_v = \left\{X_S \in \mathbb{R}^{m \times d_0} : \exists c_1 \neq c_2,\ f^{(S)}_{v,c_1}(X_S) = f^{(S)}_{v,c_2}(X_S) = \max_c f^{(S)}_{v,c}(X_S)\right\}. \tag{4}$$

Where $f^{(S)}_{v,c}(X_S)$ is the class-c logit induced at queried node $v \in S$ by the fixed local interface. This boundary captures the label-change events used by the hard-label attack.

Definition 3.5 (Dual Point). *Eq. 5 defines a dual point as an intersection between an activation boundary and a decision boundary.*

$$X^* \in \mathcal{H}_{v,j}^{(\ell)} \cap \mathcal{B}_v. \tag{5}$$

Where X^* is a queried block feature matrix, $\mathcal{H}_{v,j}^{(\ell)}$ is the activation boundary from Eq. 3, and $\mathcal{B}_v$ is the decision boundary from Eq. 4. This intersection is translated into a linear recovery equation in Section 4.

Definition 3.6 (Hard-Label Oracle). *Eq. 6 defines the label-only oracle available to the adversary in the local-query model considered here.*

$$\mathcal{O}_{hard}^{(S)}(X_S) = \left(\operatorname*{argmax}_c f_{v_1,c}^{(S)}(X_S), \ldots, \operatorname*{argmax}_c f_{v_m,c}^{(S)}(X_S) \right). \tag{6}$$

Where $\mathcal{O}_{hard}^{(S)}(X_S)$ is the hard-label response on queried block S. The oracle reveals no logits, scores, or gradients. This interface defines the only target-side access used by the main attack.

Definition 3.7 (Prediction Fidelity). *Eq. 7 defines prediction fidelity on an evaluation set $\mathcal{T}$ for the recovered surrogate used in evaluation.*

$$\text{Fidelity}(f, \hat{f} \mid \mathcal{T}) = \frac{1}{|\mathcal{T}|} \sum_{X \in \mathcal{T}} \frac{1}{n} \sum_{v=1}^{n} \mathbf{1}\left[\hat{y}_v^f(X) = \hat{y}_v^{\hat{f}}(X) \right]. \tag{7}$$

Where $\hat{y}_v^f(X)$ and $\hat{y}_v^{\hat{f}}(X)$ are the labels predicted by the target and surrogate. The quantity n is the number of nodes exposed by the evaluation interface. This metric is used throughout the experiments to measure exact functional agreement.

Definition 3.8 (Relative Error). *Eq. 8 defines relative Frobenius error between recovered and true parameter matrices used in the experiments.*

$$\text{RelErr}(\hat{W}, W) = \frac{\|\hat{W} - W\|_F}{\|W\|_F}. \tag{8}$$

Where $\text{RelErr}(\hat{W}, W)$ is the relative Frobenius error between recovered matrix $\hat{W}$ and true matrix W. Smaller values indicate closer matrix recovery under the Frobenius norm. This metric quantifies numerical parameter accuracy in the experiments section.

4 Cryptanalytic Extraction of GNNs

4.1 Attack Overview

Figure 1 summarizes the four-stage recovery chain used throughout this section. The adversary repeatedly probes one fixed ordered two-node block, so the induced local operator A_q stays constant across queries. The first stage finds dual points

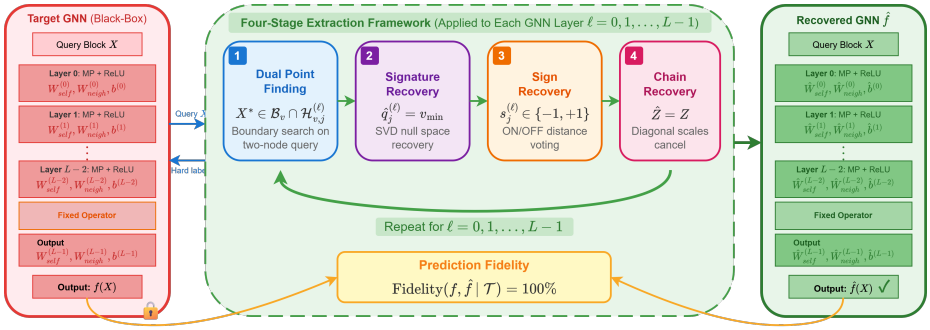

Fig. 1. Overview of the hard-label cryptanalytic extraction attack (CEA) workflow for GNNs using two queried nodes.

on class-pair decision boundaries of this fixed queried interface. The second and third stages recover signatures and resolve the remaining sign ambiguity from ON/OFF distance asymmetry. The final stage composes the recovered layers into a model $\hat{f}$, and Section 4.5 proves exact composition without explicit scale fitting.

The fixed ordered two-node interface induces one queried neuron response used throughout the recovery chain. Eq. 9 defines this affine row-column product.

$$z_{v,j}^{(\ell)}(X) = r_v^{(\ell-1)}(X)\theta_j^{(\ell)}. \tag{9}$$

Where $r_v^{(\ell-1)}(X)$ is the expanded queried row formed from the queried features, the aggregated-neighbor features, and one bias coordinate at node v. The vector $\theta_j^{(\ell)}$ is the parameter column of neuron j at layer ℓ.

4.2 Dual Point Discovery

Dual-point discovery locates intersections between the activation boundaries in Definition 3.3 and the class-pair decision boundaries. For a queried node v and classes (c_1, c_2), let $\mathcal{B}_v(c_1, c_2) \subseteq \mathcal{B}_v$ denote the corresponding node-level class-pair boundary.

The dual-finding procedure in Figure 2 is used in the first stage of the attack. The search starts from two queried inputs with different labels and uses bisection to place one anchor point on the class-pair boundary. Eq. 10 defines the midpoint update and stopping rule used by this localization step.

$$X_t = \frac{X_0 + X_1}{2}, \qquad |\Delta_{v,c_1,c_2}(X_{\text{left}})| < \epsilon. \tag{10}$$

Where X_t is the midpoint query used by bisection and ϵ is the stopping tolerance.

Walking along the localized boundary reveals bends caused by hidden activation changes in the queried interface. Eq. 11 defines the excursion point and the tangent walk used before bend detection begins.

$$\begin{aligned} X_3 &= X_{\text{left}} + \gamma R, \\ X_\alpha &= X_{\text{left}} + \alpha\,(X_4 - X_{\text{left}})\,. \end{aligned} \tag{11}$$

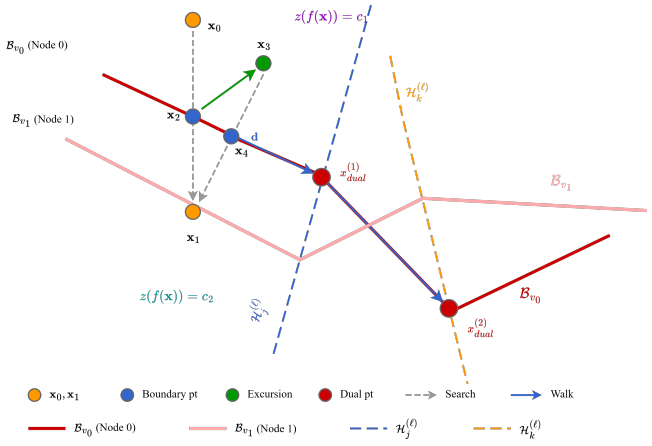

Fig. 2. Dual-point discovery by boundary search, tangent walking, bend detection, and relocation across two queried nodes.

Where γ is the excursion scale and R is a random direction matrix used during tangent walking. The point X_4 is the refined boundary point on the segment $[X_{\text{left}}, X_3]$. The point X_α is the walked boundary point parameterized by the refined displacement.

The next step detects and refines the bend on this walked path before retained-dual filtering begins. Eq. 12 defines the first detected bend location and the refined dual location obtained from the gap score $g(\alpha) = |\Delta_{v,c_1,c_2}(X_\alpha)|$.

$$\begin{aligned} \alpha^* &= \inf\{\alpha : g(\alpha) > \theta_{\text{bend}}\}, \\ \alpha_{\text{dual}} &= \operatorname*{argmin}_{\alpha \in [\alpha^* - \Delta\alpha, \alpha^*]} g(\alpha). \end{aligned} \tag{12}$$

Where θ_{bend} is the bend-detection threshold, α^* is the first detected bend location, and α_{dual} is the refined dual location.

The attack then relocates the opposite-side point by another boundary search on the same segment $[X_{\text{dual}}, X_{\text{ref}}]$. Eq. 13 defines the retained dual set passed to later recovery stages.

$$\mathcal{P}_j^{(\ell)} = \left\{ X_{\text{dual}}^{(k)} : g(\alpha_k^*) > \theta_{\text{bend}},\ X_{\text{right}}^{(k)} \in [X_{\text{dual}}^{(k)}, X_{\text{ref}}^{(k)}] \right\}. \tag{13}$$

Where $X_{\text{dual}}^{(k)}$ is the refined bend point, $g(\alpha_k^*) > \theta_{\text{bend}}$ certifies a detected bend, and $X_{\text{right}}^{(k)}$ is the relocated opposite-side anchor on the same segment. All later recovery statements are conditional on these retained dual points and the generic non-degeneracy of the stacked dual rows.

4.3 Signature Recovery

Signature recovery converts each accepted dual point into one null-space problem for the targeted hidden neuron. Proposition 4.1 first formalizes the linear relation induced by each retained dual point.

Proposition 4.1 (Dual Constraint). *Let X^* denote a dual point in the intersection $\mathcal{H}_{v,j}^{(\ell)} \cap \mathcal{B}_v(c_1, c_2)$. Eq. 14 defines the homogeneous linear constraint induced by this dual point.*

$$r_v^{(\ell-1)}(X^*)\theta_j^{(\ell)} = 0. \tag{14}$$

Where $r_v^{(\ell-1)}(X^*)$ is the expanded queried row determined by the queried representation at X^*.

Let $\hat{X}_k^{(\ell-1)} = \hat{f}^{(\ell-1)}(X_k^*) \in \mathbb{R}^{2\times d_{\ell-1}}$ denote the recovered prefix representation at the k-th dual point. Eq. 15 defines the expanded constraint row and the stacked homogeneous system for one neuron. Here $\hat{x}_{v,k}^{(\ell-1)}$ is the queried row of $\hat{X}_k^{(\ell-1)}$ at node v. The vector $\theta_j^{(\ell)}$ contains the self block, the neighbor block, and the bias.

$$\begin{aligned} r_{j,k}^{(\ell)} &= \left[\hat{x}_{v,k}^{(\ell-1)\top}\ (a_{v1}\hat{x}_{1,k}^{(\ell-1)} + a_{v2}\hat{x}_{2,k}^{(\ell-1)})^\top\ 1\right], \\ R_j^{(\ell)} &= \left[r_{j,1}^{(\ell)\top}\ \cdots\ r_{j,K}^{(\ell)\top}\right]^\top, \qquad R_j^{(\ell)}\theta_j^{(\ell)} = 0. \end{aligned} \tag{15}$$

Where $R_j^{(\ell)} \in \mathbb{R}^{K\times(2d_{\ell-1}+1)}$ is the stacked constraint matrix for neuron j at layer ℓ. Eq. 16 defines the non-degeneracy condition required by this stage for exact singular-vector recovery.

$$\operatorname{rank}\left(R_j^{(\ell)}\right) = 2d_{\ell-1}, \qquad \operatorname{rank}\left(U_j^{(\ell)}\right) = 2d_{\ell-1} - 1. \tag{16}$$

Where $R_j^{(\ell)}$ is the stacked dual matrix from Eq. 15. The matrix $U_j^{(\ell)}$ is the centered visible matrix defined below in Eq. 17. The first rank condition fixes the visible rank before centering. The second rank condition leaves one null direction after centering.

Centering removes the bias coordinate before singular-vector recovery of the hidden-neuron direction. Let $u_{j,k}^{(\ell)}$ denote the feature part of $r_{j,k}^{(\ell)}$ without the bias coordinate, and let $\mu_j^{(\ell)} = \frac{1}{K}\sum_{k=1}^{K} u_{j,k}^{(\ell)}$ be its empirical mean. Eq. 17 defines the centered matrix used in the SVD stage.

$$U_j^{(\ell)} = \begin{bmatrix} u_{j,1}^{(\ell)} - \mu_j^{(\ell)} \\ \vdots \\ u_{j,K}^{(\ell)} - \mu_j^{(\ell)} \end{bmatrix} = P\Sigma V^\top. \tag{17}$$

Where $\mu_j^{(\ell)}$ is the empirical mean of the feature rows and $P\Sigma V^\top$ is the singular value decomposition of the centered matrix.

Proposition 4.2 (SVD Recovery). *If the centered rows span all but one direction, the smallest right singular vector of $U_j^{(\ell)}$ recovers the neuron signature up to sign. Eq. 18 defines the recovered singular-vector direction used by the attack.*

$$\hat{q}_j^{(\ell)} = v_{\min} = V_{:,\,2d_{\ell-1}}. \tag{18}$$

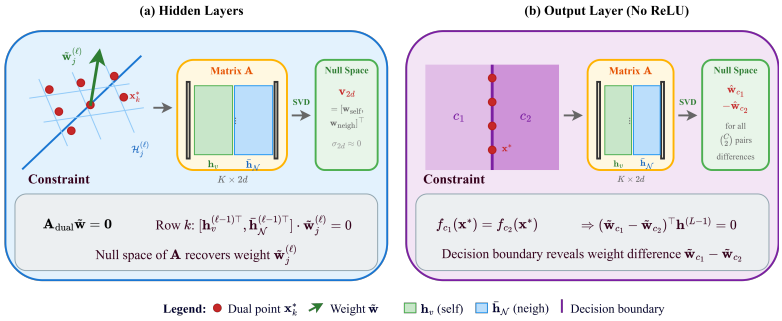

Fig. 3. SVD-based signature recovery for hidden layers and the output layer from boundary-derived linear constraints.

Where $v_{\min}$ *is the right singular vector associated with the smallest singular value of* $U_j^{(\ell)}$.

The recovered direction then splits into self and neighbor blocks of the same hidden-neuron signature. Eq. 19 defines the singular-value ratio test used to retain stable signature estimates.

$$r_{\mathrm{sv}} = \frac{\sigma_{\min}}{\sigma_{\min-1}}, \qquad r_{\mathrm{sv}} < \tau_{\mathrm{sv}}^{(\ell)}. \tag{19}$$

Where r_{sv} is the singular-value ratio, $\sigma_{\min}$ and $\sigma_{\min-1}$ are the two smallest singular values, and $\tau_{\mathrm{sv}}^{(\ell)}$ is the layer acceptance threshold. Smaller ratios indicate a sharper one-dimensional null space and therefore more stable signature estimates.

Figure 3 summarizes the shared centered null-space pipeline for hidden and output signatures. Hidden layers recover neuron directions from stacked dual rows, whereas the output layer chains class-pair differences into class parameters.

Bias recovery then reuses the uncentered dual equations obtained during the same accepted singular-vector stage. Let $\hat{w}_{\mathrm{self},j}^{(\ell)}$ and $\hat{w}_{\mathrm{neigh},j}^{(\ell)}$ denote the two recovered blocks of $\hat{q}_j^{(\ell)}$. Let $\hat{a}_{v,k}^{(\ell-1)} = a_{v1}\hat{x}_{1,k}^{(\ell-1)} + a_{v2}\hat{x}_{2,k}^{(\ell-1)}$ denote the recovered aggregated neighbor feature at node v. Eq. 20 defines the final bias estimate after median aggregation across retained dual rows.

$$\hat{b}_j^{(\ell)} = \mathrm{median}_{k=1}^{K}\left(-\hat{w}_{\mathrm{self},j}^{(\ell)\top}\hat{x}_{v,k}^{(\ell-1)} - \hat{w}_{\mathrm{neigh},j}^{(\ell)\top}\hat{a}_{v,k}^{(\ell-1)}\right). \tag{20}$$

Where $\hat{a}_{v,k}^{(\ell-1)}$ is the recovered neighbor aggregate at node v. The median suppresses unstable dual rows before the final bias estimate is fixed. This completes the same signature-recovery stage as Eq. 17 and Eq. 18. Appendix Algorithm 1 records the corresponding executable hidden-layer routine for this bias-aggregation step.

Later layers require an additional reliability screen because prefix errors can perturb some dual rows. Eq. 21 defines the residual score and percentile cutoff used before the final SVD. Rows with residual score above τ_{res} are discarded

before the final null-space estimate.

$$\begin{aligned} r_k &= \left| U_j^{(\ell)}[k,:]\hat{q}_j^{(\ell)} \right|, \\ \tau_{\text{res}} &= \text{Percentile}_{80}\left(\{r_k\}_{k=1}^{K}\right). \end{aligned} \tag{21}$$

Where r_k is the residual score of the k-th dual row and τ_{res} is the fixed 80th-percentile cutoff. The thresholds ϵ, θ_{bend}, $\tau_{\text{sv}}^{(\ell)}$, and τ_{res} are fixed numerical tolerances that reject unstable boundary events or ill-conditioned null spaces.

The same stage also recovers the output layer from class-pair boundaries rather than from hidden activation hyperplanes alone. Each accepted class-pair boundary contributes one homogeneous row constraint on the corresponding class-difference vector. Let $\Delta\hat{\Theta}^{(L-1)}$ denote the stacked class-difference vectors. Let $\hat{\Theta}^{(L-1)}$ denote the stacked class-parameter vectors. Eq. 22 defines the chain system used to recover class parameters from consecutive differences.

$$\Delta\hat{\Theta}^{(L-1)} = \begin{bmatrix} I_p & -I_p & 0 & \cdots & 0 \\ 0 & I_p & -I_p & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & I_p & -I_p \end{bmatrix} \hat{\Theta}^{(L-1)}. \tag{22}$$

Where $\Delta\theta_{c_1,c_2}^{(L-1)}$ is the difference vector $\theta_{c_1}^{(L-1)} - \theta_{c_2}^{(L-1)}$. The quantity $p = 2d_{L-1}+1$ is the visible dimension of one class parameter. The matrix I_p is the $p \times p$ identity. The block difference matrix links consecutive class-pair differences across the recovered output parameters. Appendix Algorithm 2 records the corresponding executable routine for exact output-layer signature recovery.

4.4 Sign Recovery

Sign recovery resolves the sign ambiguity that remains after null-space recovery of the hidden and output directions. The activation normal is computed on the recovered surrogate rather than on the target service. Eq. 23 defines the recovered activation normal used for local probing.

$$\hat{n}_j^{(\ell)}(X^*) = \frac{\nabla_X z_{v,j}^{(\ell)}(X)\big|_{X=X^*}}{\left\| \nabla_X z_{v,j}^{(\ell)}(X)\big|_{X=X^*} \right\|_2}. \tag{23}$$

Where $\hat{n}_j^{(\ell)}(X^*)$ is the unit normal of the recovered activation boundary at the selected dual point. It is obtained from the recovered prefix map and the recovered neuron signature.

Sign comparison then uses the local decision-boundary tangent estimated at the same dual point on the target boundary. Let $g(X^*)$ denote the hard-label estimate of the local decision normal obtained from nearby boundary

perturbations. Eq. 24 defines its normalized form and projected tangent direction.

$$\begin{aligned}\hat{m}(X^*) &= \frac{g(X^*)}{\|g(X^*)\|_2},\\ \hat{D}_t &= \frac{(I-\hat{n}_j^{(\ell)}(X^*)\hat{n}_j^{(\ell)}(X^*)^\top)\hat{m}(X^*)}{\left\|(I-\hat{n}_j^{(\ell)}(X^*)\hat{n}_j^{(\ell)}(X^*)^\top)\hat{m}(X^*)\right\|_2}.\end{aligned} \tag{24}$$

Where $\hat{m}(X^*)$ is the normalized decision normal obtained without logits or target gradients. The direction $\hat{D}_t$ is its projection onto the tangent subspace orthogonal to $\hat{n}_j^{(\ell)}(X^*)$.

Eq. 25 defines the first boundary-crossing distances measured on the ON and OFF sides of the same dual point. Both distances are measured by repeated hard-label queries and bisection on the first observed label change along the ray. This comparison supplies the local orientation evidence used by sign recovery.

$$\begin{aligned}&X_{\mathrm{ON}} = X^* + \delta\hat{n}_j^{(\ell)}(X^*), \qquad X_{\mathrm{OFF}} = X^* - \delta\hat{n}_j^{(\ell)}(X^*),\\ &d_{\mathrm{side}} = \inf\left\{\alpha > 0 : \hat{y}(X_{\mathrm{side}} + \alpha\hat{D}_t) \neq \hat{y}(X_{\mathrm{side}})\right\}.\end{aligned} \tag{25}$$

Where δ is the normal perturbation size and d_{side} is the first crossing distance on the chosen side.

Proposition 4.3 (Distance Asymmetry and Orientation). *Assume that X^* is single-differential and all other neurons keep fixed activation states in a small neighborhood. Eq. 26 defines the resulting sign from the measured asymmetry.*

$$s_j^{(\ell)} = \begin{cases} +1 & \text{if } d_{OFF} > d_{ON},\\ -1 & \text{if } d_{ON} > d_{OFF}.\end{cases} \tag{26}$$

Where d_{ON} is the first tangent crossing distance from X_{ON}, and d_{OFF} is the corresponding distance from X_{OFF}.

Inside a small neighborhood, the local margin is affine on each side of the activation boundary. The active side carries the hinge contribution of neuron j, so it reaches the next decision boundary first along the tangent ray. Eq. 27 defines the confidence-weighted aggregation applied across multiple dual points.

$$v_j^{(\ell)} = \sum_{k=1}^{K_{\mathrm{sign}}} c_j^{(\ell,k)} s_j^{(\ell,k)}, \qquad s_j^{(\ell)} = \mathrm{sign}(v_j^{(\ell)}). \tag{27}$$

Where $c_j^{(\ell,k)} = |d_{\mathrm{OFF}}^{(k)} - d_{\mathrm{ON}}^{(k)}|/(d_{\mathrm{OFF}}^{(k)} + d_{\mathrm{ON}}^{(k)})$ is the confidence weight, and K_{sign} is the number of voting points. If nearby boundaries violate local isolation, the corresponding probes receive low confidence or are discarded before aggregation. The final vote therefore depends only on hard-label boundary localization, finite-difference fitting, and first-crossing tests. Appendix Algorithms 3 and 4 record the corresponding probing and confidence-weighted voting routines for hidden-layer and output-layer sign resolution.

Figure 4 illustrates Eq. 26 for one recovered neuron. The side with the smaller tangent crossing distance fixes the local orientation used in the final vote.

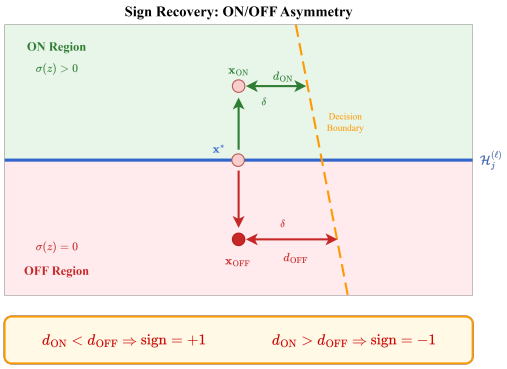

Fig. 4. Sign recovery by comparing ON-side and OFF-side boundary distances around a dual point.

4.5 Scale Independence via Chain Recovery

Sequential recovery leaves each hidden layer scaled by a positive diagonal matrix inherited from the previous stage. Prior extraction pipelines usually fit these scales explicitly before continuing, but the reused prefix carries the same diagonal ambiguity forward here. The next theorem shows that this chain cancels at the output layer.

Theorem 4.1 (Scale Chain Cancellation). *Let $D^{(\ell)} \in \mathbb{R}^{d_\ell \times d_\ell}$ be a positive diagonal matrix with $D^{(0)} = I_{d_0}$ and $D^{(L)} = I_{d_L}$. Eq. 28 defines the recovered layer matrix under sequential recovery.*

$$\hat{\Theta}^{(\ell)} = \bar{D}^{(\ell)} \Theta^{(\ell)} D^{(\ell+1)-1}, \qquad \bar{D}^{(\ell)} = \begin{bmatrix} D^{(\ell)} & 0 & 0 \\ 0 & D^{(\ell)} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{28}$$

We note that the first two diagonal blocks rescale the self and neighbor channels by the same incoming layer scale. The recovered prefix satisfies $\hat{X}^{(\ell)} = X^{(\ell)} D^{(\ell)-1}$, and $D^{(L)} = I_{d_L}$ because the output layer is measured in the original logit coordinates. The final logits therefore satisfy $\hat{Z} = Z$.

Proof. The base case is exact because the queried input is observed, so $\hat{X}^{(0)} = X^{(0)}$ and $\hat{\bar{X}}^{(0)} = \bar{X}^{(0)}$. Assume inductively that $\hat{X}^{(\ell)} = X^{(\ell)} D^{(\ell)-1}$. Eq. 29 defines the lifted queried representation under the same diagonal factor.

$$\hat{\bar{X}}^{(\ell)} = \bar{X}^{(\ell)} \bar{D}^{(\ell)-1}. \tag{29}$$

Where $\bar{D}^{(\ell)}$ acts on the two visible feature blocks and preserves the bias column.

Eq. 30 defines the propagated scaled logits obtained by substituting this identity into the recovered layer matrix.

$$\hat{Z}^{(\ell+1)} = \hat{\bar{X}}^{(\ell)} \hat{\Theta}^{(\ell)} = \bar{X}^{(\ell)} \bar{D}^{(\ell)-1} \bar{D}^{(\ell)} \Theta^{(\ell)} D^{(\ell+1)-1} = Z^{(\ell+1)} D^{(\ell+1)-1}. \tag{30}$$

Positive diagonal scaling commutes with ReLU columnwise, so the induction propagates to layer $\ell+1$. Eq. 31 defines the output-layer cancellation of the same chain.

$$\hat{Z} = \bar{X}^{(L-1)}\bar{D}^{(L-1)-1}\bar{D}^{(L-1)}\Theta^{(L-1)}D^{(L)-1} = \bar{X}^{(L-1)}\Theta^{(L-1)} = Z. \quad (31)$$

Where $D^{(L)} = I_{d_L}$ because the output layer is measured in the original logit coordinates.

This cancellation removes the explicit layer-scale fitting used by prior cryptanalytic pipelines [7,4,6]. Proposition 4.4 records the resulting conditional complexity claim used throughout the paper and experiments.

Proposition 4.4 (Conditional Polynomial Complexity). *Under the accepted single-differential, local-isolation, and non-degeneracy assumptions, the recovery pipeline has polynomial post-processing and query complexity. Eq. 32 defines the resulting asymptotic bound for the attack pipeline.*

$$T_{\text{post}} = O\big(NKd^2 + NK_{\text{sign}}\big), \qquad Q = \text{poly}(L, d, K, C, \kappa_{\text{dual}}). \quad (32)$$

Where $N = \sum_{\ell=0}^{L-1} d_\ell$ is the number of recovered neurons and K is the accepted dual coverage per neuron. The quantity K_{sign} is the number of sign probes per neuron, and κ_{dual} summarizes the query cost of accepted dual discovery.

This proposition isolates the cost of post-processing accepted duals from the geometric difficulty of finding them. The polynomial-time statement is conditioned on the fixed local-query model and the geometric assumptions stated above.

5 Experimental Evaluation

5.1 Experimental Setup

All experiments were conducted on Ubuntu 22.04 using a single NVIDIA RTX 4090 GPU. Unless otherwise noted, the detailed recovery, fidelity, and distillation results use a dedicated retrained 32-32 $\times$ 4-4 teacher checkpoint. That checkpoint reaches 96.52%, 92.68%, and 92.26% accuracy on the train, validation, and test splits, whereas Table 3 reports a separate architecture sweep.

5.2 Parameter and Sign Recovery

Table 2 summarizes layer-wise relative parameter error, sign confidence, and sign accuracy after sequential composition. Relative error follows Definition 3.8 and the RelErr convention in Eq. 8. Every recovered layer remains numerically stable, and the same results show exact sign voting across all 68 neurons.

Table 2. Layer-wise summary of relative parameter error and sign recovery.

		Relative Error			Sign Confidence		
Layer	Neurons	W_{self}	W_{neigh}	bias	Median	Mean	Accuracy
0	16	3.35×10^{-7}	4.61×10^{-7}	1.66×10^{-8}	0.699	0.665	100%
1	16	5.55×10^{-7}	5.54×10^{-7}	1.59×10^{-7}	0.653	0.643	100%
2	16	1.23×10^{-6}	1.09×10^{-6}	3.40×10^{-7}	0.729	0.652	100%
3	16	1.99×10^{-6}	1.74×10^{-6}	6.89×10^{-7}	0.499	0.494	100%
4	4	1.71×10^{-13}	1.63×10^{-13}	4.85×10^{-14}	0.971	0.865	100%
Total	68	-	-	-	-	-	100%

The hidden-layer mean relative errors stay below 2×10^{-6} after sequential recovery, and the output layer reaches near machine precision. Table 2 also shows 100% sign accuracy in every layer, so the recovered blocks preserve the exact functional chain required by Theorem 4.1. Figure 5 shows the clearest ON/OFF separation in Layers 0 and 4, whereas Layer 3 remains closer to the diagonal while still supporting exact sign voting.

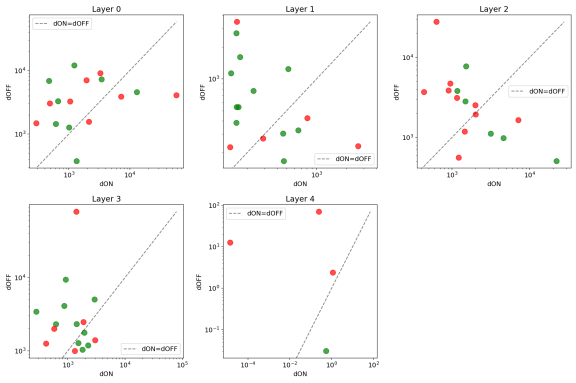

Fig. 5. Scatter plot of d_{on} and d_{off} for sign recovery.

5.3 Output Fidelity Analysis

The final surrogate composes the recovered signatures, biases, and signs in one pipeline. Definition 3.7 measures exact label agreement in this evaluation, and Section 4.5 proves that no explicit scale-fitting stage is required. The recovered model attains **100% fidelity** on all 8,171 evaluated nodes in the evaluation interface. This result verifies Theorem 4.1 and shows that numerical recovery errors do not accumulate across layers.

The logits distribution in Figure 6 remains tightly concentrated near zero across evaluated queries. The maximum difference is 3.90×10^{-5}, and the mean difference is 1.64×10^{-6}, which confirms exact class agreement with only microscopic numeric discrepancy.

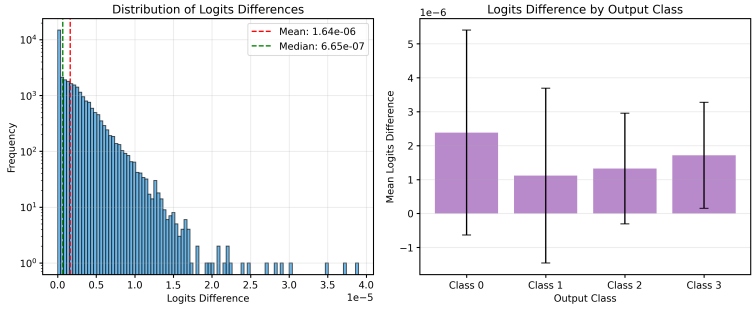

Fig. 6. Distribution of logits differences between original and recovered models.

5.4 Generalization Experiments

Table 3 extends the evaluation from the four-class ogbn-arxiv subset [17] to four-class Cora and CiteSeer subsets from the standard Planetoid benchmarks [31,25]. The sweep still includes wider and deeper ogbn-arxiv targets up to the 32-32 $\times$ 5-4 model with 10,660 parameters. It therefore tests whether exact recovery reflects one benchmark-specific configuration or the shared message-passing ReLU structure.

Table 3. Generalization analysis.

Dataset	Nodes	Classes	Architecture	Avg.	Max.	Fidelity
ogbn-arxiv	8171	4	32-32 $\times$ 3-4	3.80×10^{-7}	5.87×10^{-6}	100.00%
ogbn-arxiv	8171	4	32-24 $\times$ 4-4	1.47×10^{-7}	4.96×10^{-6}	100.00%
ogbn-arxiv	8171	4	32-32 $\times$ 4-4	1.21×10^{-6}	4.01×10^{-5}	100.00%
ogbn-arxiv	8171	4	32-16 $\times$ 5-4	1.30×10^{-6}	4.55×10^{-5}	100.00%
ogbn-arxiv	8171	4	32-16 $\times$ 4-4	9.42×10^{-7}	3.20×10^{-5}	100.00%
ogbn-arxiv	8171	4	32-32 $\times$ 5-4	1.96×10^{-6}	6.66×10^{-5}	100.00%
Cora	1804	4	32-32 $\times$ 3-4	8.50×10^{-7}	2.89×10^{-5}	100.00%
CiteSeer	2223	4	32-32 $\times$ 3-4	1.10×10^{-6}	3.74×10^{-5}	100.00%

Every listed checkpoint reaches 100% end-to-end fidelity under the same recovery chain. Across ogbn-arxiv architectures, average error stays between 1.47×10^{-7} and 1.96×10^{-6}, while maximum error stays between 4.96×10^{-6} and 6.66×10^{-5}. The same 32-32 $\times$ 3-4 pipeline also reaches 100% fidelity on Cora and CiteSeer with average error below 1.10×10^{-6} and maximum error below 3.74×10^{-5}. These results support a limited but meaningful notion of generality for the present CEA setting.

5.5 Comparison with Knowledge Distillation

Table 4 compares exact recovery with knowledge distillation. Here "Original" denotes the dedicated retrained 32-32 $\times$ 4-4 teacher checkpoint. "Random Fid." measures agreement on 10,000 standard-normal two-node queries from the attack domain.

The comparison isolates exact extraction from approximate imitation on the queried local domain. The strongest baseline, Hard-Random 8K, reaches 99.82%

Table 4. Exact recovery versus distillation.

Method	Queries	Train Acc	Test Acc	Train Fid.	Test Fid.	Random Fid.
Original	—	96.52%	92.26%	—	—	—
Hard-Train 1K	8.17M	87.70%	83.05%	90.12%	83.42%	54.23%
Hard-Train 5K	40.86M	95.35%	93.06%	98.12%	95.52%	59.98%
Hard-Random 1K	8.17M	93.67%	92.26%	95.82%	94.41%	58.36%
Hard-Random 8K	65.37M	96.41%	91.89%	99.82%	99.63%	61.95%
Soft-Train 1K	8.17M	94.04%	91.34%	96.05%	94.84%	62.39%
Soft-Train 5K	40.86M	26.03%	40.97%	26.41%	40.05%	17.40%
Soft-Random 1K	8.17M	88.50%	82.80%	90.73%	84.89%	54.43%
Soft-Random 8K	65.37M	89.15%	82.86%	91.61%	86.36%	60.29%
Ours	**48.68M**	**96.52%**	**92.26%**	**100.00%**	**100.00%**	**100.00%**

train fidelity and 99.63% test fidelity, yet only 61.95% random fidelity, whereas our method reaches 100% on all fidelity metrics.

5.6 Countermeasure Analysis

CEA suggests two countermeasure directions. One breaks the deterministic boundary conditions required by exact recovery, and the other constrains the large adaptive query budget.

Theoretical Countermeasures. Noise-aware training and inference-time jitter are natural theoretical defenses because Eq. 14 and Eq. 25 both rely on deterministic local responses across repeated identical queries. Randomized boundary crossings would obstruct null-space accumulation and sign voting, although the perturbation level must still preserve task accuracy [15,21].

Practical Countermeasures. API-side defenses should target the large adaptive query footprint required by CEA. Full exact recovery on the main target uses 48.68M node queries, so rate limiting, authenticated access, per-client quotas, duplicate-query suppression, and audit alarms on repeated fixed-block probes can raise the attack cost substantially and make the pattern operationally visible [28,21].

6 Conclusion

This paper shows that parameter security fails for message-passing ReLU GNNs under a strong hard-label local-query adversary. The recovery chain converts dual-point geometry into hidden signatures, hidden signs, and output parameters, while the scale-chain theorem explains exact layer composition without explicit scale fitting. Together, these results answer the paper's main question within the fixed local-query model studied here.

Experiments confirm this result across the evaluated ogbn-arxiv, Cora, and CiteSeer four-class targets, including the largest tested 32-32 $\times$ 5-4 architecture. Every recovered model reaches 100% prediction fidelity with small average and maximum errors. The analysis remains specific to hard-label message-passing ReLU GNNs under the two-node interface, which leaves broader query models and stronger stochastic or access-control defenses for future work.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (U2336209) and the Guangdong Basic and Applied Basic Research Foundation (2022A1515140090).

Disclosure of Interests. The authors have no competing interests to declare.

References

1. Addepalli, S., Nayak, G.K., Chakraborty, A., Radhakrishnan, V.B.: Degan: Data-enriching gan for retrieving representative samples from a trained classifier. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 3130–3137 (2020). https://doi.org/10.1609/aaai.v34i04.5709
2. Asselineau, R., Derbez, P., Fouque, P.A., Minaud, B.: Cryptanalytic extraction of deep neural networks with non-linear activations. Cryptology ePrint Archive (2026), https://eprint.iacr.org/2026/253
3. Biton Dor, M., Mirsky, Y.: Efficient model extraction via boundary sampling. In: Proceedings of the 2024 Workshop on Artificial Intelligence and Security. p. 1–11. AISec '24, Association for Computing Machinery (2024). https://doi.org/10.1145/3689932.3694756
4. Canales-Martínez, I.A., Chávez-Saab, J., Hambitzer, A., Rodríguez-Henríquez, F., Satpute, N., Shamir, A.: Polynomial time cryptanalytic extraction of neural network models. In: Advances in Cryptology – EUROCRYPT 2024. pp. 3–33. Springer Nature Switzerland (2024). https://doi.org/10.1007/978-3-031-58734-4_1
5. Canales-Martínez, I.A., Santos, D.: Extracting some layers of deep neural networks in the hard-label setting. Cryptology ePrint Archive, Paper 2025/1118 (2025), https://eprint.iacr.org/2025/1118
6. 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: Advances in Cryptology – EUROCRYPT 2025. pp. 364–396. Springer Nature Switzerland (2025). https://doi.org/10.1007/978-3-031-91107-1_13
7. Carlini, N., Jagielski, M., Mironov, I.: Cryptanalytic extraction of neural network models. In: Advances in Cryptology – CRYPTO 2020. pp. 189–218. Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-56877-1_7
8. Carlini, N., Jagielski, M., Mironov, I.: Cryptanalytic extraction of neural network models. In: Annual International Cryptology Conference. pp. 189–218. Springer (2020)
9. Chandrasekaran, V., Chaudhuri, K., Giacomelli, I., Jha, S., Yan, S.: Exploring connections between active learning and model extraction. In: Proceedings of the 29th USENIX Conference on Security Symposium. SEC'20, USENIX Association (2020). https://doi.org/10.5555/3489212.3489286
10. Chen, Y., Dong, X., Guo, J., Shen, Y., Wang, A., Wang, X.: Hard-label cryptanalytic extraction of neural network models. In: Advances in Cryptology – ASIACRYPT 2024. p. 207–236. Springer-Verlag (2024). https://doi.org/10.1007/978-981-96-0944-4_7
11. Chen, Y., Dong, X., Ma, R., Shen, Y., Wang, A., Yu, H., Wang, X.: Delving into cryptanalytic extraction of prelu neural networks. In: Advances in Cryptology – ASIACRYPT 2025. Springer (2025)

12. Chen, Z., Tang, S., Gao, Z., Su, Y., Qin, L., Dong, X.: Algebraic attack on convolutional neural network with max pooling. Cryptology ePrint Archive (2026), https://eprint.iacr.org/2026/241
13. Correia-Silva, J.R., Berriel, R.F., Badue, C., de Souza, A.F., Oliveira-Santos, T.: Copycat cnn: Stealing knowledge by persuading confession with random non-labeled data. In: 2018 International Joint Conference on Neural Networks (IJCNN). pp. 1–8 (2018). https://doi.org/10.1109/IJCNN.2018.8489592
14. Foerster, H., Mullins, R., Shumailov, I., Hayes, J.: Beyond slow signs in high-fidelity model extraction. In: Advances in Neural Information Processing Systems. vol. 37 (2024). https://doi.org/10.48550/arXiv.2406.10011
15. Fredrikson, M., Jha, S., Ristenpart, T.: Model inversion attacks that exploit confidence information and basic countermeasures. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. p. 1322–1333. CCS '15, Association for Computing Machinery (2015). https://doi.org/10.1145/2810103.2813677
16. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems (2017)
17. Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., Leskovec, J.: Open graph benchmark: Datasets for machine learning on graphs. Advances in neural information processing systems (2020)
18. Jagielski, M., Carlini, N., Berthelot, D., Kurakin, A., Papernot, N.: High accuracy and high fidelity extraction of neural networks. In: Proceedings of the 29th USENIX Conference on Security Symposium. SEC'20, USENIX Association (2020). https://doi.org/10.5555/3489212.3489288
19. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2017)
20. Liu, H., Siproudhis, A., Experton, S., Lorenz, P., Boura, C., Peyrin, T.: Navigating the deep: End-to-end extraction on deep neural networks. arXiv preprint arXiv:2506.17047 (2025), https://arxiv.org/abs/2506.17047
21. Oliynyk, D., Mayer, R., Rauber, A.: I know what you trained last summer: A survey on stealing machine learning models and defences. ACM Comput. Surv. **55**(14s) (2023). https://doi.org/10.1145/3595292
22. Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z.B., Swami, A.: Practical black-box attacks against machine learning. In: Proceedings of the 2017 ACM on Asia conference on computer and communications security. pp. 506–519 (2017)
23. Qi, X., Lei, H., Wei, L., Sun, X., Wang, M.: Cryptanalytic extraction of neural networks with various activation functions. Cryptology ePrint Archive (2026), https://eprint.iacr.org/2026/178
24. Rolnick, D., Kording, K.P.: Identifying weights and architectures of unknown relu networks (2019), arXiv preprint arXiv:1910.00744
25. Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., Eliassi-Rad, T.: Collective classification in network data. AI Magazine **29**(3), 93–106 (2008)
26. Shen, Y., He, X., Han, Y., Zhang, Y.: Model stealing attacks against inductive graph neural networks. In: IEEE Symposium on Security and Privacy (2022)
27. Sun, X., Lei, H., Wei, L., Qi, X., Hu, K., Wang, M., Wang, W.: Cryptanalytic extraction of convolutional neural networks. Cryptology ePrint Archive (2026), https://eprint.iacr.org/2026/139
28. Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., Ristenpart, T.: Stealing machine learning models via prediction apis. In: 25th USENIX Security Symposium (USENIX Security 16). pp. 601–618. USENIX Association (2016)

29. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: International Conference on Learning Representations (2018)
30. Wei, L., Lei, H., Qi, X., Sun, X., Gao, L., Hu, K., Wang, W., Wang, M.: Cryptanalytic extraction of recurrent neural network models. Cryptology ePrint Archive (2026), https://eprint.iacr.org/2026/168
31. Yang, Z., Cohen, W.W., Salakhutdinov, R.: Revisiting semi-supervised learning with graph embeddings. In: Proceedings of the 33rd International Conference on Machine Learning. pp. 40–48 (2016)

A Appendix

This appendix records the compact pseudocode used in the attack.

A.1 Pseudocode

Algorithms 1 and 3 apply to input and hidden layers. Algorithms 2 and 4 apply to the output layer. All four routines use the same minimal two-node local block.

Algorithm 1 Hidden Signature Recovery

In: $\mathcal{P}_j^{(\ell)}$, node v, prefix $\hat{f}^{(\ell-1)}$, threshold $\tau_{\mathrm{sv}}^{(\ell)}$.
Out: $\hat{q}_j^{(\ell)}$, $\hat{b}_j^{(\ell)}$, or rejection.
for each $X_k^* \in \mathcal{P}_j^{(\ell)}$ **do**
    Form $[\hat{x}_{v,k}^\top, \hat{a}_{v,k}^\top, 1]$ from $\textsc{PrefixRepr}(\hat{f}^{(\ell-1)}, X_k^*)$ and append it to $R_j^{(\ell)}$.
end for
Center $R_j^{(\ell)}$ and compute $U_j^{(\ell)} = P\Sigma V^\top$.
$\hat{q}_j^{(\ell)} \leftarrow V_{:,\mathrm{last}}$ and $r_{\mathrm{sv}} \leftarrow \sigma_{\min}/\sigma_{\min-1}$.
if $r_{\mathrm{sv}} \geq \tau_{\mathrm{sv}}^{(\ell)}$ **then**
    return rejection
end if
Split $\hat{q}_j^{(\ell)}$ into self and neighbor blocks.
Set $\hat{b}_j^{(\ell)} \leftarrow \mathrm{median}\Big(\{-\hat{w}_{\mathrm{self},j}^{(\ell)\top}\hat{x}_{v,k} - \hat{w}_{\mathrm{neigh},j}^{(\ell)\top}\hat{a}_{v,k}\}_{k=1}^{K}\Big)$.
return $(\hat{q}_j^{(\ell)}, \hat{b}_j^{(\ell)})$

Algorithm 2 Output Signature Recovery

In: $\{\mathcal{B}_{c_1,c_2}\}_{c_1<c_2}$, prefix $\hat{f}^{(L-1)}$, class count C.
Out: $\{\Delta\hat{\theta}^{(L-1)}_{c_1,c_2}\}$ and $\{\hat{\theta}^{(L-1)}_c\}_{c=0}^{C-1}$.
for each pair (c_1, c_2) with $0 \le c_1 < c_2 < C$ **do**
  for each $X_k^* \in \mathcal{B}_{c_1,c_2}$ **do**
    Form $[\hat{x}_{v,k}^\top, \hat{a}_{v,k}^\top, 1]$ from $\textsc{PrefixRepr}(\hat{f}^{(L-1)}, X_k^*)$ and append it to $A^{(c_1,c_2)}$.
  end for
  Center $A^{(c_1,c_2)}$, compute $U\Sigma V^\top$, and set $\Delta\hat{\theta}^{(L-1)}_{c_1,c_2} \leftarrow V_{:,\text{last}}$.
end for
Set $\hat{\theta}_0^{(L-1)} \leftarrow 0$.
for $c = 1$ to $C-1$ **do**
  Recover $\hat{\theta}_c^{(L-1)}$ with Eq. 22.
end for
return $(\{\Delta\hat{\theta}^{(L-1)}_{c_1,c_2}\}, \{\hat{\theta}^{(L-1)}_c\}_{c=0}^{C-1})$

Algorithm 3 Hidden Sign Recovery

In: $\mathcal{P}_j^{(\ell)}$, $\hat{q}_j^{(\ell)}$, oracle $\hat{y}(\cdot)$, radius δ.
Out: $s_j^{(\ell)}$ and $\text{conf}_j^{(\ell)}$.
Set $v_j^{(\ell)} \leftarrow 0$ and $w_j^{(\ell)} \leftarrow 0$.
for each $X_k^* \in \mathcal{P}_j^{(\ell)}$ **do**
  Estimate $\hat{n}_j^{(\ell)}(X_k^*)$, estimate the decision normal, and form $\hat{D}_t$ by Eq. 24.
  Set $X_{\text{ON}} \leftarrow X_k^* + \delta\hat{n}_j^{(\ell)}(X_k^*)$ and $X_{\text{OFF}} \leftarrow X_k^* - \delta\hat{n}_j^{(\ell)}(X_k^*)$.
  Query d_{ON} and d_{OFF} along $\hat{D}_t$.
  Compute $s_j^{(\ell,k)}$ by Eq. 26 and $c_j^{(\ell,k)}$ by Eq. 27.
  Update $v_j^{(\ell)} \leftarrow v_j^{(\ell)} + c_j^{(\ell,k)} s_j^{(\ell,k)}$ and $w_j^{(\ell)} \leftarrow w_j^{(\ell)} + c_j^{(\ell,k)}$.
end for
$s_j^{(\ell)} \leftarrow \text{sign}(v_j^{(\ell)})$ and $\text{conf}_j^{(\ell)} \leftarrow |v_j^{(\ell)}|/(w_j^{(\ell)} + 10^{-12})$.
return $(s_j^{(\ell)}, \text{conf}_j^{(\ell)})$

Algorithm 4 Output Sign Recovery

In: $\mathcal{B}_c$, $\hat{\theta}_c^{(L-1)}$, oracle $\hat{y}(\cdot)$, radius δ.
Out: $s_c^{(L-1)}$ and $\text{conf}_c^{(L-1)}$.
Set $v_c^{(L-1)} \leftarrow 0$ and $w_c^{(L-1)} \leftarrow 0$.
$\hat{n}_c^{(L-1)} \leftarrow \hat{\theta}_c^{(L-1)}/\|\hat{\theta}_c^{(L-1)}\|_2$.
for each $X_k^* \in \mathcal{B}_c$ **do**
  Estimate the decision normal and form $\hat{D}_t$ by Eq. 24.
  Set $X_+ \leftarrow X_k^* + \delta\hat{n}_c^{(L-1)}$ and $X_- \leftarrow X_k^* - \delta\hat{n}_c^{(L-1)}$.
  Query d_+ and d_- along $\hat{D}_t$.
  Set the local sign $s_c^{(L-1,k)} \leftarrow +1$ if $d_- > d_+$, else $s_c^{(L-1,k)} \leftarrow -1$.
  Set the weight $c_c^{(L-1,k)} \leftarrow |d_- - d_+|/(d_- + d_+ + 10^{-12})$.
  Update $v_c^{(L-1)} \leftarrow v_c^{(L-1)} + c_c^{(L-1,k)} s_c^{(L-1,k)}$ and $w_c^{(L-1)} \leftarrow w_c^{(L-1)} + c_c^{(L-1,k)}$.
end for
$s_c^{(L-1)} \leftarrow \text{sign}(v_c^{(L-1)})$ and $\text{conf}_c^{(L-1)} \leftarrow |v_c^{(L-1)}|/(w_c^{(L-1)} + 10^{-12})$.
return $(s_c^{(L-1)}, \text{conf}_c^{(L-1)})$