Security of Infocommunication Systems and Internet of Things

2026 Vol 4, No 1

https://doi.org/10.31861/sisiot2026.1.01004

Received 26 February 2026; revised 02 April 2026; accepted 05 June 2026; published 30 June 2026

Neural Network Configuration of the r-Adding Walk in Pollard's Rho Method Using the Spectral Gap for the Elliptic Curve Discrete Logarithm Problem

Mykola Onai* **and Danylo Hulko**

Department of Computer Systems Software, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine

*Corresponding author (E-mail: onay@pzks.fpm.kpi.ua)

ABSTRACT This paper investigates two neural-network formulations for configuring the r-adding walk in Pollard's Rho method for the elliptic curve discrete logarithm problem on short Weierstrass curves over prime fields. The first formulation uses an NM-r classifier to predict the increment-table size r and an NM-inc regression network to generate the normalized increment coefficients (α_i, β_i) directly from the curve and subgroup context. An approximate spectral-gap dataset with context-grouped training, validation, and test partitions was constructed for this formulation. Its evaluation revealed severe target-class imbalance for NM-r and instability of the single-target increment labels under higher-precision transition sampling, which limited reliable end-to-end generalization. These findings motivated a second formulation in which a neural model scores a finite pool of candidate increment tables and selects the candidate with the highest predicted spectral quality for a fixed r. To evaluate the second formulation without sampling uncertainty in the target, an exact dataset containing 5,000 curve contexts and 480,000 candidate configurations was generated for r ∈ {8, 16, 32}, with the prime field modulus p and subgroup order N satisfying $2^{15} < p, N < 2^{20}$. Each target was computed from an exact r × r tag-transition matrix obtained by enumerating every subgroup state. Candidate-scoring models were trained using context-grouped 80/10/10 splitting, evaluated across multiple input representations and random seeds, and frozen before integration into a C# implementation of Pollard's Rho method. The benchmark compared neural selection, deterministic random selection, an L = 512 sampled-gap oracle, and an exact-gap oracle on 500 held-out contexts using 30 paired trials for each context and r, producing 180,000 successful measurements. Across r ∈ {8, 16, 32}, neural selection reduced the mean number of iterations normalized by √N by 2.52% relative to random selection, with a context-bootstrap 95% confidence interval of [0.24%, 4.79%] and a paired sign-flip value of p = 0.0325. For r = 8, the reduction was 5.25% with a 95% confidence interval of [1.40%, 9.00%] and a Holm-adjusted value of p = 0.0237 across the three neural-versus-random tests. No conclusive improvement was obtained for r = 16 or r = 32. Moreover, improvements in the exact spectral gap showed only a weak association with reductions in the number of iterations of Pollard's Rho method, and the exact-gap oracle did not provide a consistent advantage. The results demonstrate a modest empirical benefit of neural candidate selection in the evaluated research-scale setting while showing that the spectral gap alone is not a universal predictor of the performance of Pollard's Rho method.

KEYWORDS software engineering, elliptic curve discrete logarithm problem, Pollard's Rho method, r-adding walk, neural networks.

I. INTRODUCTION

Elliptic curve cryptography relies on the presumed intractability of the elliptic curve discrete logarithm problem in groups of points defined over finite fields [1, 2]. This work considers a short Weierstrass curve over the prime field $\mathbb{F}_p$ given by

$$E(\mathbb{F}_p): y^2 \equiv x^3 + ax + b \pmod{p}, \quad (1)$$

where $a, b \in \mathbb{F}_p$, and $4a^3 + 27b^2 \not\equiv 0 \pmod{p}$.

Let P be a generator of a cyclic subgroup of $E(\mathbb{F}_p)$ of order N and let Q be another subgroup element. The elliptic curve discrete logarithm problem is the task of recovering the unknown scalar k from the relation:

$$Q = kP. \quad (2)$$

The hardness of Eq. 2 underpins the security of widely deployed elliptic curve protocols, therefore advances in generic attacks remain directly relevant to parameter selection and implementation practice [2].

Pollard's Rho method is a fundamental baseline for the elliptic curve discrete logarithm problem cryptanalysis because it applies to arbitrary cyclic groups and requires limited memory, while providing an expected running time that scales as:

$$T(N) = O(\sqrt{N}) \quad (3)$$

in group operations for a subgroup of order N [3]. Although Eq. 3 captures the asymptotic behavior, practical efficiency depends on how the iteration function is instantiated. In particular, the r-adding walk construction introduces design choices such as the walk size r and the increment selection mechanism, which influence the mixing properties of the induced Markov process and therefore affect the expected time to reach a collision in concrete settings [4, 5].

A persistent difficulty is that r-adding walk parameters are often chosen by manual tuning or heuristics, which can be inconsistent across curve contexts and can obstruct

systematic methodology. Motivated by the need for a systematic configuration procedure, this paper investigates whether neural-network-based parameter selection can improve the behavior of an r-adding walk in Pollard's Rho method when the spectral gap of the induced tag-transition model is used as the configuration-quality criterion. Two complementary formulations are considered. The first is a direct-prediction formulation in which the NM-r classifier predicts the increment-table size r, while the NM-inc regression network generates a normalized coefficient table $\{(\alpha_i, \beta_i)\}_{i=0}^{r-1}$ from the curve and subgroup context. The second is a candidate-scoring formulation in which a neural model evaluates a finite pool of increment tables for a fixed r and selects the candidate with the highest predicted spectral quality. The objectives of the study are to construct reproducible datasets for both formulations, evaluate label stability and model generalization on previously unseen curve contexts, compare the learned selection policies with random and oracle-based baselines, and determine empirically whether the selected increment tables reduce the number of iterations required by Pollard's Rho method.

The main contributions of this study are as follows. First, a reproducible approximate-label dataset pipeline is defined for the direct-prediction formulation based on NM-r and NM-inc, including deterministic context generation, canonical serialization of coefficient tables, context-grouped data splitting, and higher-precision analysis of target-label stability. Second, an exact-label dataset generation procedure is introduced for the candidate-scoring formulation, in which the spectral quality of every candidate is computed from an exact $r \times r$ tag-transition matrix obtained by enumerating all states of the corresponding prime-order subgroup. Third, the study provides an empirical comparison of the direct-prediction and candidate-scoring formulations, including classification, regression, candidate-ranking, representation-ablation, and multi-seed stability metrics on previously unseen curve contexts. Fourth, the selected increment tables are integrated into a C# implementation of Pollard's Rho method and evaluated in a paired experiment against deterministic random selection, an $L = 512$ sampled-gap oracle, and an exact-gap oracle. Finally, the complete experimental pipeline records dataset configurations, random seeds, model checkpoints, source hashes, raw measurements, and statistical-analysis parameters to support independent reproduction of the reported results.

The remainder of this paper is organized as follows. Section II reviews recent research on Pollard's Rho method, r-adding walks, spectral analysis of finite-state transition models, and machine-learning-based cryptographic parameter selection. Section III describes the approximate-label and exact-label datasets used to evaluate the two neural configuration formulations and defines the corresponding spectral-quality measures. Section IV presents the direct-prediction formulation based on the NM-r and NM-inc networks. Section V introduces the candidate-scoring formulation, its input representations, training procedure, and candidate-selection policy. Section VI describes the integration of both formulations into Pollard's Rho method and defines the paired experimental benchmark. Section VII reports the dataset, neural-model, spectral-selection, and Pollard's Rho method results. Section VIII discusses the implications, limitations, and scalability of the proposed approaches. Finally, Section IX summarizes the conclusions and directions for further research.

II. ANALYSIS OF RECENT RESEARCH AND PUBLICATIONS

Pollard's Rho method remains a canonical generic approach for solving the elliptic curve discrete logarithm problem on the short Weierstrass curve in Eq. (1), where the goal is to recover k from $Q = kP$ in a cyclic subgroup of order N [1-3, 7]. The method builds a pseudo-random sequence in the group while tracking a linear representation of each visited point. A typical state at iteration i is written as:

$$R_i = a_i P + b_i Q, \quad a_i, b_i \in \mathbb{Z}_N,$$

and a collision $R_u = R_v$ is exploited to obtain an algebraic relation among coefficients. Under standard non-degeneracy conditions, the discrete logarithm can be recovered by modular inversion

$$k \equiv (a_u - a_v)(b_v - b_u)^{-1} \pmod{N},$$

which highlights why the walk must induce collisions while avoiding coefficient patterns that make the inversion undefined.

Within this framework, the r-adding walk is a widely studied family of iteration functions that constructs the next state by adding one of r precomputed increments selected by a tagging rule [4-5]. Let the increment set be $\{M_1, \ldots, M_r\}$ and let $j(i) \in \{1, \ldots, r\}$ be determined from a tag derived from R_i. The update can be expressed as:

$$R_{i+1} = R_i + M_{j(i)},$$
$$M_j = \alpha_j P + \beta_j Q,$$
$$(a_{i+1}, b_{i+1}) \equiv (a_i + \alpha_{j(i)}, b_i + \beta_{j(i)}) \pmod{N},$$

so the practical behavior depends jointly on r, the distribution of tags across the group, and the algebraic diversity of (α_j, β_j). Prior analyses emphasize that poor choices can lead to slow mixing, short cycles, or biased visitation of subsets of states, which delays collisions even though the asymptotic attack model remains unchanged. As a consequence, parameter selection is not a cosmetic implementation detail but a central determinant of whether the walk approximates the heuristic assumption of randomness that underlies the standard Rho argument.

The behavior of an r-adding walk can be studied through a finite-state model of its tag dynamics. Let X_t denote the subgroup point visited at iteration t, and let $h(X_t) \in \{0, \ldots, r-1\}$ be its tag. The reduced tag process is represented by a transition matrix $P \in [0, 1]^{r \times r}$, whose elements are defined as:

$$P_{ij} = \Pr(h(X_{t+1}) = j \mid h(X_t) = i).$$

The sequence of tags is treated as an aggregated Markov approximation of the underlying walk rather than as a complete representation of the point-level process. Let the eigenvalues of P be ordered so that $\lambda_1 = 1$ is the stationary eigenvalue and:

$$|\lambda_2| \geq |\lambda_2| \geq \ldots \geq |\lambda_r|.$$

The spectral gap of the reduced transition model is defined as:

$$\gamma(P) = 1 - \max_{2 \leq k \leq r} |\lambda_k(P)|$$

For a finite ergodic Markov chain, the magnitude of the non-stationary eigenvalues is related to the rate at which the chain converges toward its stationary distribution [13]. Spectral analysis has also been used to connect random-walk mixing and collision bounds for Pollard's Rho method [8]. Consequently, a larger value of $\gamma(P)$ generally indicates faster mixing of the modeled tag process and weaker persistence of tag correlations. However, this theoretical relationship applies to the reduced Markov model and does not by itself prove that an increment table with a larger spectral gap will require fewer iterations in Pollard's Rho method. The practical relationship between the spectral criterion and the collision behavior of the complete elliptic-curve walk must therefore be evaluated experimentally.

Recent studies demonstrate increasing interest in applying machine-learning and optimization techniques to elliptic-curve cryptography, including cryptographic parameter selection, protocol-level optimization, and implementation improvement [6, 9-12]. These works indicate that data-driven methods can support the exploration of large parameter spaces and can complement conventional analytical and implementation-oriented optimization. Nevertheless, most existing studies focus on curve parameters, protocol configurations, or arithmetic implementations rather than on the iteration function of Pollard's Rho method. In particular, they do not provide a reproducible methodology for learning the increment-table configuration of an r-adding walk from curve and subgroup contexts or for evaluating the resulting configuration through both a spectral criterion and a direct collision-search experiment.

Despite these advances, the literature does not provide an experimentally validated and reproducible framework for neural configuration of an r-adding walk in Pollard's Rho method. In particular, previous work does not compare direct generation of increment-table parameters with neural scoring of candidate tables, analyze the stability of spectral-gap-based training targets, or evaluate whether neural selection reduces the number of iterations required to recover a discrete logarithm. The present study addresses this gap by examining both direct prediction and candidate scoring under context-grouped evaluation and by testing the selected configurations in an integrated Pollard's Rho method implementation.

III. DATASET GENERATION AND SPECTRAL EVALUATION

The experimental study uses two complementary datasets corresponding to the two investigated neural configuration formulations. The datasets are generated independently and are not combined during model training or evaluation. The first dataset supports direct prediction through NM-r and NM-inc. It contains curve and subgroup contexts over prime fields satisfying $2^{25} \leq p < 2^{30}$, with candidate configurations generated for $r \in \{8, 16, 32, 64\}$. Candidate quality is estimated from sampled tag transitions using $L = 512$ observations per transition-matrix row. This dataset is used to examine whether the increment-table size and normalized coefficient table can be predicted directly from the task context.

The second dataset supports neural scoring and selection of candidate increment tables. To remove sampling uncertainty from the target spectral-gap values, it uses research-scale prime fields and prime-order subgroups satisfying $2^{15} \leq p, N < 2^{20}$. Candidate configurations are generated for $r \in \{8, 16, 32\}$, and their quality is computed from exact $r \times r$ tag-transition matrices obtained by enumerating every state of the corresponding subgroup. The reduced field and subgroup range makes exhaustive construction of the tag-transition model computationally feasible and is not intended to represent cryptographic-strength parameters.

Both datasets use deterministic generation seeds, canonical serialization of candidate tables, and context-grouped training, validation, and test partitions. All records associated with the same elliptic-curve context are assigned to exactly one partition. This prevents information leakage between correlated candidate configurations and ensures that model evaluation measures generalization to previously unseen curve and subgroup contexts.

A. Approximate-gap dataset for direct prediction. The approximate-gap dataset contains 5,000 independently generated elliptic-curve contexts. A stored context is represented as:

$$C = (p, a, b, N, P, Q),$$

where p is the prime field modulus, a and b are the coefficients of the short Weierstrass curve defined by Eq. (1), N is the prime order of the selected subgroup, P is its generator, and $Q = kP$ is the target point instantiated using a hidden validation scalar $k \in \{1, \ldots, N-1\}$. The prime modulus satisfies:

$$2^{25} \leq p < 2^{30},$$

while the subgroup-selection procedure requires:

$$N \geq 2^{20}.$$

The scalar k is retained only for mathematical validation of the generated context and is not included in the neural-network inputs or used as a learning target.

Prime moduli are generated by rejection sampling. An integer is sampled uniformly from $[2^{25}, 2^{30})$, it is made odd, and it is tested for primality using a deterministic primality procedure suitable for this bit range, then the process is repeated until a prime is obtained. For a fixed prime p, coefficients a and b are sampled independently and uniformly from $\{0, \ldots, p-1\}$, after which the curve is accepted only if it is non-singular. Nonsingularity is enforced through the discriminant computed modulo p,

and the acceptance condition is:

$$-16\left(4a^3+27b^2\right) \not\equiv 0 \pmod{p}.$$

If the condition fails, coefficients are resampled while keeping p fixed.

After a valid curve is obtained, a prime order subgroup is selected and a generator is constructed through group operations. The group cardinality $\left|E\left(\mathbb{F}_p\right)\right|$ is computed, it is factored, and a prime divisor N is chosen as the subgroup order under a fixed exclusion rule that removes trivial cases by requiring $N \geq 2^{20}$. If no such prime divisor exists, the entire curve is discarded and a new context is sampled. Given $\left|E\left(\mathbb{F}_p\right)\right|$ and N, a subgroup generator is derived from a random curve point. A point $R \in E\left(\mathbb{F}_p\right)$ is obtained by sampling an $x \in \{0, \ldots, p-1\}$, evaluating the right-hand side x^3+ax+b modulo p, checking quadratic residuosity, and constructing a corresponding y when a square root exists, then repeating until a valid point is found. The generator candidate is then computed as $P = \dfrac{\left|E\left(\mathbb{F}_p\right)\right|}{N}R$. This construction forces P into the subgroup of order dividing N, and since N is prime, it suffices to reject the identity point. The generator is verified by checking $P \neq \mathcal{O}$ and $NP = \mathcal{O}$, where $\mathcal{O}$ denotes the point at infinity. Contexts that yield $P = \mathcal{O}$ are rejected by resampling R.

For each accepted context, a target point Q is constructed by drawing an integer x uniformly from $\{1, \ldots, N-1\}$ and setting $Q = xP$. The value x is used only to instantiate candidate r-adding increments through the point Q, and it is not used as a direct learning label. This separation ensures that the learning objective depends on configuration quality rather than on recovering a discrete logarithm value.

Candidate configurations are generated for $r \in \{8, 16, 32, 64\}$. For every context and every admissible value of r, four independent coefficient tables are generated, resulting in 16 candidates per context and 80,000 candidates in the complete dataset. A candidate table is represented as:

$$T_r^{(c)} = \left\{\left(\alpha_i^{(c)}, \beta_i^{(c)}\right)\right\}_{i=0}^{r-1},$$

where the candidate index is $c \in \{0, 1, 2, 3\}$. The coefficients are sampled independently and uniformly from $\mathbb{Z}_N$, subject to:

$$\left(\alpha_i^{(c)}, \beta_i^{(c)}\right) \neq (0, 0).$$

Each accepted table is canonicalized by lexicographically sorting its coefficient pairs. The resulting order defines the final index assignment and is used consistently during increment construction, transition evaluation, serialization, and target generation. The corresponding increment points are computed as:

$$M_i^{(c)} = \alpha_i^{(c)}P + \beta_i^{(c)}Q.$$

The partition index is defined through a deterministic tagging function h that maps a group element to one of the r indices using the least significant bits of the x-coordinate. Since every candidate uses an r that is a power of two, one writes $r = 2^k$ and fixes:

$$h(X) = \mathrm{lsb}_k(x(X)).$$

Here $x(X)$ is the x-coordinate of X represented as an integer in $\{0, \ldots, p-1\}$, and lsb_k extracts the k least significant bits and returns an integer in $\{0, \ldots, r-1\}$. In the implementation this mapping is evaluated by a bit mask operation that depends only on k and the canonical representative of $x(X)$, which makes h fully deterministic for a given context and candidate configuration.

Each candidate receives an approximate quality label derived from an empirical tag-transition matrix. For a fixed context and r, transition counters $C^{(c)} = \left[C_{ij}^{(c)}\right]_{i,j=0}^{r-1}$ are initialized for all four candidate tables. Starting states are shared across candidates with the same context and r. At every sampling step, a scalar s is selected uniformly from $\mathbb{Z}_N$, and the corresponding subgroup point is computed as $X = sP$. Its current tag is determined as

$$i = h_r(X).$$

For each candidate c, the transition associated with row i is evaluated by computing

$$X'^{(c)} = X + M_i^{(c)}$$

and

$$j^{(c)} = h_r\left(X'^{(c)}\right).$$

The observation is then accumulated as:

$$C_{ij^{(c)}}^{(c)} \leftarrow C_{ij^{(c)}}^{(c)} + 1.$$

Sampling continues until every row contains exactly L observations. Separate rejection-sampling loops are not executed for individual rows. Since the current tag depends only on the shared starting point X and the fixed value of r, same accepted starting states contribute to the corresponding rows of all four candidate matrices. In the main dataset:

$$L = 512.$$

The estimated transition probabilities are computed as:

$$\widehat{P}_{ij}^{(c)} = \frac{C_{ij}^{(c)}}{L}.$$

Therefore, every row of $\widehat{P}_{ij}^{(c)}$ sums to one, and all candidate tables belonging to the same context and r are evaluated using identical sampled starting states.

Let the eigenvalues of the empirical transition matrix $\widehat{P}^{(c)}$ be ordered so that $\lambda_1^{(c)} = 1$ is the stationary eigenvalue. The approximate spectral-gap reward of candidate c is defined as

$$\widehat{\gamma}^{(c)} = 1 - \max_{2 \leq k \leq r} \left|\lambda_k^{(c)}\right|.$$

For every context, all 16 candidate configurations are compared, and the target candidate is selected as:

$$c^* = \operatorname*{argmax}_c \widehat{\gamma}^{(c)}.$$

If multiple candidates have numerically identical rewards, the candidate identifier and candidate index are used as a deterministic tie-breaking rule. The table size

associated with c^* becomes the classification target for NM-r. The normalized coefficient table of the same candidate becomes the regression target for NM-inc:

$$y_{\text{inc}} = \left(\frac{\alpha_0}{N}, \frac{\beta_0}{N}, \ldots, \frac{\alpha_{r-1}}{N}, \frac{\beta_{r-1}}{N} \right).$$

Thus, both direct-prediction targets are derived from the same approximate spectral-selection rule.

To assess the stability of the approximate labels, a high-precision reevaluation is performed on 50 contexts, corresponding to 10% of the held-out test contexts. The original curve contexts and all 16 candidate tables are retained without modification. Their transition matrices are reevaluated using $L_{\text{HP}} = 4096$ observations per row and an independently recorded random seed. The reevaluation measures the correlation between the rewards obtained with $L = 512$ and $L_{\text{HP}} = 4096$, as well as the proportion of contexts for which both settings select the same target candidate and the same value of r. These stability results are reported in Section VII rather than being used to modify the frozen training labels.

B. Exact-gap dataset for candidate scoring. A separate dataset was generated to investigate whether a neural model can learn to rank candidate increment tables according to their spectral properties. Unlike the approximate-gap dataset, this dataset contains the spectral gap of the complete reduced tag-transition matrix for every candidate. Exact enumeration required smaller problem instances; therefore, the field modulus and subgroup order were restricted to $2^{15} \le p < 2^{20}$ and $2^{15} \le N < 2^{20}$.

The dataset contains 5000 independently generated curve contexts. For each context and each value $r \in \{8, 16, 32\}$, 32 candidate coefficient tables were generated. Consequently, the dataset contains

$$5000 \times 3 \times 32 = 480000$$

labelled candidates. Each candidate is represented by its context parameters, the value of r, and the normalized coefficients α_i / N and β_i / N, where:

$$M_i = \alpha_i P + \beta_i Q, \quad i \in \{0, \ldots, r-1\}.$$

For every candidate table, all subgroup states were enumerated. For a subgroup state $X_s = sP$, where $s \in \{0, \ldots, N-1\}$, the current tag and the tag after applying the corresponding increment were computed as:

$$i = h_r(X_s),$$
$$j = h_r(X_s + M_i).$$

The observation was added to the transition count C_{ij}. After all N subgroup states had been processed, the exact reduced transition probabilities were obtained as:

$$P_{ij} = \frac{Cj}{\sum_{k=0}^{r-1} C_{ik}}.$$

Thus, each row was normalized by the actual number of enumerated subgroup states assigned to the corresponding tag. The exact spectral gap of the reduced transition matrix was then calculated as:

$$\gamma(P) = 1 - \max_{2 \le k \le r} |\lambda_k(P)|,$$

where the eigenvalues are ordered so that $\lambda_1(P) = 1$.

The term "exact" in this experiment refers to the complete enumeration of the subgroup when constructing the $r \times r$ tag-transition matrix. It does not imply the construction or eigen decomposition of an $N \times N$ state-transition matrix. The reduced representation is retained because the investigated spectral criterion is defined for the tag process induced by the r-adding walk.

In addition to the exact matrix, sampled transition matrices were constructed using $L \in \{16, 32, 64, 128, 512\}$. Nested samples were used so that the observations included at a smaller L were also included at every larger sampling level. These matrices were retained as candidate features and approximation baselines, whereas the exact spectral gap obtained by complete subgroup enumeration remained the regression target.

The contexts were deterministically partitioned into 4000 training, 500 validation, and 500 test contexts. All candidates associated with a particular curve context were assigned to the same subset. An explicit validation confirmed that no context identifier occurred in more than one subset. Any secret scalar used internally to accelerate offline subgroup construction was excluded from the exported neural-network features.

For each pair of a curve context and r, the candidate with the largest exact spectral gap was treated as the finite-pool oracle:

$$c^* = \operatorname*{argmax}_{c \in C_r} \gamma\left(P^{(c)}\right).$$

This oracle was used only as an evaluation reference. The candidate-scoring model was trained to predict the spectral quality of every individual candidate rather than to reproduce the coefficient vector of a single selected table. At inference time, the predicted scores can therefore be used to rank independently generated candidate tables and return the highest-ranked table to Pollard's Rho method.

IV. DIRECT NEURAL PARAMETER GENERATION

The first investigated approach formulates the configuration of the r-adding walk as direct supervised parameter prediction. It consists of two neural networks with separate roles. The NM-r network predicts the increment-table size from the curve and subgroup context, whereas the NM-inc network predicts the coefficient table conditioned on the selected value of r. The admissible table sizes are restricted to the ordered set

$$\mathfrak{R} = \{8, 16, 32, 64\}.$$

NM-r is formulated as a four-class classifier whose output identifies one element of $\mathfrak{R}$. Its training label is the value of r associated with the candidate configuration having the largest estimated spectral gap in the approximate-gap dataset described in Section III-A.

NM-inc is formulated as a masked regression model. Given the same context and a one-hot representation of r, it predicts the normalized coefficients:

$$\left(\frac{\alpha_0}{N}, \frac{\beta_0}{N}, \ldots, \frac{\alpha_{r-1}}{N}, \frac{\beta_{r-1}}{N}\right)$$

of the selected increment table. The corresponding increment points are subsequently constructed as:

$$M_i = \alpha_i P + \beta_i Q, \quad i \in \{0, \ldots, r-1\}.$$

To support every admissible value of r with a fixed network architecture, the NM-inc output dimension is set to:

$$M2r_{\max} = 128, \quad r_{\max} = 64,$$

Only the first $2r$ output components participate in loss computation and deterministic decoding. The remaining $128 - 2r$ components are masked and do not affect training or inference. This direct-generation formulation is evaluated as the initial neural configuration strategy. Its limitations motivate the candidate-scoring formulation introduced in Section V.

Input features are constructed deterministically from the curve and subgroup parameters. The NM-r input is the five-dimensional vector:

$$z = \left[\log_2 p, a/p, b/p, \log_2 N, N/p\right],$$

where p is the prime modulus, a and b are the coefficients of the short Weierstrass curve, and N is the order of the subgroup generated by P. Before feature construction, the curve coefficients are represented by their canonical integer residues:

$$a, b \in \{0, \ldots, p-1\}.$$

No dataset-dependent min-max scaling, standardization, or z-score normalization is applied. The ratios $a/p, b/p$, and N/p provide relative representations of the corresponding integer values, whereas $\log_2(p)$ and $\log_2(N)$ retain information about the absolute field and subgroup scales.

For NM-inc, the selected table size is represented by the one-hot vector:

$$e_r = onehot_{[8, 16, 32, 64]}(r)$$

whose component order corresponds exactly to $[8, 16, 32, 64]$.

The NM-inc input is therefore defined as:

$$z_{\text{inc}} = \text{concat}(z, e_r) \in \mathbb{R}^9$$

This representation conditions NM-inc on the value predicted by NM-r while preserving a fixed input dimension. In the direct-generation experiment, the coordinates of P and Q are not included in z_{inc}. The implications of this restricted representation for predicting context-specific increment coefficients are examined in Sections VII and VIII.

Both NM-r and NM-inc are multilayer perceptrons with three hidden layers containing 256 units each. Every hidden layer applies a rectified linear unit followed by dropout with probability 0.1. For an input vector x, the shared form of the hidden backbone is:

$$h_1 = \text{Dropout}\left(\text{ReLU}\left(W_1 x + b_1\right), 0.1\right),$$
$$h2 = \text{Dropout}\left(\text{ReLU}\left(W_2 x + b_2\right), 0.1\right),$$
$$h_3 = \text{Dropout}\left(\text{ReLU}\left(W_3 x + b_3\right), 0.1\right).$$

For NM-r, the input dimension is 5, and the output layer produces four unnormalized logits:

$$\ell = W_r h_3 + b_r \in \mathbb{R}^4.$$

The predicted class is determined as:

$$k = \underset{k \in \{0, 1, 2, 3\}}{\text{argmax}} \ \ell_k,$$

and is mapped to the corresponding element of the ordered set $[8, 16, 32, 64]$.

Softmax probabilities may be calculated when class probabilities are required for analysis, but the training loss is applied directly to the logits.

For NM-inc, the input dimension is 9, and the output layer contains 128 components followed by a sigmoid activation:

$$\hat{y} = \sigma\left(W_{\text{inc}} h_3 + b_{\text{inc}}\right) \in (0, 1)^{128}.$$

The output represents normalized coefficient pairs up to the maximum supported table size $r_{\max} = 64$. For a training or inference instance with table size r, only the prefix: $\hat{y}_{1:2r}$ is used. This masking rule prevents the unused output components from contributing to either the regression loss or coefficient decoding.

The NM-inc output is converted into an integer coefficient table by deterministic decoding. Let $\hat{y}$ denote the sigmoid output of NM-inc. For the selected table size r, the first $2r$ components are interpreted as ordered coefficient pairs. For every $i \in \{0, \ldots, r-1\}$, decoding is performed as:

$$\hat{\alpha}_i = \text{round}\left(N\hat{y}_{2i}\right) \mod N,$$
$$\hat{\beta}_i = \text{round}\left(N\hat{y}_{2i+1}\right) \mod N.$$

Consequently,

$$\hat{\alpha}_i, \hat{\beta}_i \in \{0, \ldots, N-1\}.$$

A deterministic postprocessing procedure is then applied to ensure that the decoded coefficient table satisfies the validity requirements of the r-adding walk. First, the pair corresponding to the point at infinity is prohibited. If:

$$\left(\hat{\alpha}_i, \hat{\beta}_i\right) = (0, 0),$$

it is replaced with:

$$\left(\hat{\alpha}_i, \hat{\beta}_i\right) = (1, 0).$$

Second, duplicate coefficient pairs are removed. If the current pair has already occurred among the accepted pairs, its second component is incremented modulo N:

$$\hat{\beta}_i \leftarrow \left(\hat{\beta}_i + 1\right) \mod N.$$

This operation is repeated until the resulting pair is unique. If it produces $(0, 0)$, the increment is continued until both the nonzero and uniqueness constraints are satisfied.

Finally, the coefficient pairs are sorted lexicographically to obtain a canonical serialized representation:

$$\left(\hat{\alpha}_0, \hat{\beta}_0\right) \le_{\text{lex}} \cdots \le_{\text{lex}} \left(\hat{\alpha}_{r-1}, \hat{\beta}_{r-1}\right).$$

The corresponding increment points are constructed after postprocessing:

$$\hat{M}_i = \hat{\alpha}_i P + \hat{\beta}_i Q, \quad i \in \{0, \ldots, r-1\}.$$

The same decoding and canonicalization rules are applied to every model output. This procedure guarantees a syntactically valid and reproducibly serialized increment table, but it does not by itself guarantee that the repaired table preserves the spectral quality of the raw network output. Accordingly, empirical use of a decoded table requires an additional spectral evaluation. In the present study, the direct NM-inc formulation was not promoted to that stage after the target-stability check reported in Section VII-B.

NM-r and NM-inc are trained as separate supervised models. For NM-r, cross-entropy loss is applied directly to the four output logits. Let $\ell \in \mathbb{R}^4$ be the logits and let $y_r \in \{0, 1, 2, 3\}$ be the target class index. The classification loss is:

$$\mathcal{L}_r = -\log\left(\frac{\exp\left(\ell_{y_r}\right)}{\sum_{k=0}^{3}\exp\left(\ell_k\right)}\right).$$

The primary experiment uses the unweighted loss. Because the selected r labels are imbalanced, an additional experiment applies class weights calculated from the class frequencies in the training subset:

$$\mathcal{L}_r^{(\omega)} = -\omega_{y_r}\log\left(\frac{\exp\left(\ell_{y_r}\right)}{\sum_{k=0}^{3}\exp\left(\ell_k\right)}\right).$$

The weighted and unweighted models are evaluated separately and are not selected using the test subset.

For NM-inc, the regression target is the normalized coefficient vector:

$$y_{\text{inc}} = \left[\frac{\alpha_0}{N}, \frac{\beta_0}{N}, \ldots, \frac{\alpha_{r-1}}{N}, \frac{\beta_{r-1}}{N},\right].$$

SmoothL1 loss is calculated only over the first $2r$ output components. For a scalar residual d, it is defined as:

$$\text{SmoothL1}(d) = \begin{cases} \frac{1}{2}\text{d}^2, & |\text{d}| < 1, \\ |\text{d}| - \frac{1}{2}, & |\text{d}| \geq 1. \end{cases}$$

The masked NM-inc objective is:

$$\mathcal{L}_{\text{inc}} = \frac{1}{2r}\sum_{j=0}^{2r-1}\text{SmoothL1}\left(\hat{y}_j - y_j\right).$$

Components with $j \geq 2r$ do not participate in either loss calculation or gradient propagation for the corresponding training instance.

Both networks are optimized with AdamW using learning rate $\eta = 10^{-3}$ and weight decay $\lambda = 10^{-4}$. The minibatch size is 1024, and training proceeds for at most 50 epochs. Early stopping is applied when the validation loss does not improve for 5 consecutive epochs. The checkpoint with the lowest validation loss is retained. Seeds for dataset splitting, model initialization, and minibatch ordering are fixed and stored in the experiment manifests.

The approximate-gap dataset is split by complete curve contexts into 80% training, 10% validation, and 10% test subsets. Thus, the 5000 contexts are divided into 4000, 500, and 500 contexts, respectively. Every candidate and selected target associated with one context remains in exactly one subset. An explicit leakage check verifies that the sets of context identifiers are pairwise disjoint.

NM-r is evaluated using classification accuracy, macro-averaged F1-score, per-class recall, and the confusion matrix. Its results are additionally compared with a majority-class baseline. For NM-inc, masked mean absolute error over the active $2r$ components and the integrity rate of decoded tables were defined as planned evaluation metrics. However, the high-precision label-stability check reported in Section VII-B showed that the single-table targets were not sufficiently stable. Consequently, NM-inc was not promoted to a definitive downstream evaluation, and no coefficient-regression performance claim is made.

V. NEURAL CANDIDATE SCORING AND SELECTION

The limitations of direct coefficient prediction motivated a second formulation in which the neural network does not generate an increment table directly. Instead, a deterministic candidate generator produces a finite pool $C_r = \left\{C_r^{(1)}, \ldots, C_r^{(K)}\right\}$ for a fixed value of r. A neural surrogate independently estimates the spectral gap of every candidate and selects:

$$\hat{c} = \underset{c \in C_r}{\text{argmax}}\ \hat{\gamma}_\theta\left(C_r^{(c)} \mid X\right),$$

where $X = (p, a, b, N, P, Q)$ is the curve context and $\hat{\gamma}_\theta$ is the spectral-gap estimate produced by the trained model. Candidate generation uses a recorded pseudorandom seed and does not require the exact transition matrix during inference.

A. Candidate representations. The candidate-scoring experiments use the nine-dimensional context vector:

$$x_{\text{ctx}} = \left[\log_2(p), \frac{a}{p}, \frac{b}{p}, \log_2(N), \frac{N}{p}, \frac{x_P}{p}, \frac{y_P}{p}, \frac{x_Q}{p}, \frac{y_Q}{p}\right]$$

Unlike the direct NM-inc formulation, this representation explicitly includes the normalized affine coordinates of P and Q. Context features are standardized using means and standard deviations calculated exclusively from the training subset.

Several candidate representations are compared in an ablation study. The context-only representation contains no candidate-specific information and is used as a negative-control baseline:

$$x_{\text{context}} = x_{\text{ctx}}.$$

The coefficient representation concatenates the context with the normalized coefficient table:

$$x_{\text{coeff}} = concat\left(x_{\text{ctx}}, \frac{\alpha_0}{N}, \frac{\beta_0}{N}, \ldots, \frac{\alpha_{r-1}}{N}, \frac{\beta_{r-1}}{N}\right).$$

Its dimension is $9 + 2r$. The point representation uses the affine coordinates, tag, and point-at-infinity indicator of each increment M_i:

$$m_i = \left[\frac{x_{M_i}}{p}, \frac{y_{M_i}}{p}, \frac{h_r\left(M_i\right)}{r-1}, \mathbb{I}\left[M_i = \mathcal{O}\right]\right],$$

$$x_{\text{points}} = \text{concat}\left(x_{\text{ctx}}, m_0, \ldots, m_{r-1}\right).$$

Its dimension is $9 + 4r$.

Two sampled-matrix representations are also evaluated. For $L \in \{128, 512\}$, the sampled transition counts are divided by L and flattened:

$$x_{\text{matrix},L} = \text{concat}\left(x_{\text{ctx}}, vec\left(\frac{C^{(L)}}{L}\right)\right).$$

The corresponding input dimension is $9 + r^2$.

The sampled matrices are candidate features rather than training targets. In every case, the target is the exact reduced-matrix spectral gap obtained by complete subgroup enumeration.

B. Scoring architectures. For the context, coefficient, point, and flattened-matrix representations, the scorer is a multilayer perceptron with three hidden layers of 256 units. Each hidden layer uses ReLU followed by dropout with probability 0.1. The final layer returns one scalar:

$$\hat{\gamma}_\theta^{(c)} = f_\theta\left(x^{(c)}\right).$$

The scalar is interpreted as the predicted spectral gap of candidate c. All 32 candidates belonging to the same training context and fixed r are processed as one candidate group.

An additional matrix-aware architecture processes the $L = 512$ representation without flattening its row structure. For every tag row, the sampled transition values are concatenated with the corresponding coefficient and increment-point features. The resulting row vector is projected into a 64-dimensional embedding. A two-layer Transformer encoder with four attention heads and a feed-forward dimension of 128 processes the sequence of r tag rows. Mean pooling is then applied, the pooled representation is concatenated with x_{ctx}, and a regression head produces the predicted spectral gap.

These architectures estimate candidate quality rather than reproducing an oracle coefficient vector. At inference time, the candidate with the largest predicted score is returned as the neural configuration of the r-adding walk.

C. Training objective and selection metrics. Separate candidate-scoring models are trained for $r = 8$, $r = 16$, and $r = 32$. Each training instance is a group containing all $K = 32$ candidates associated with the same curve context and fixed r. This grouping allows the objective to account for both absolute spectral-gap prediction and relative candidate ordering.

The exact spectral gaps are standardized using the mean μ_γ and standard deviation σ_γ calculated exclusively from the training subset:

$$t_{g,c} = \frac{\gamma_{g,c} - \mu_\gamma}{\sigma_\gamma},$$

where g identifies the context group and $c \in \{1, \ldots, K\}$ identifies a candidate. Let $s_{g,c}$ be the standardized score predicted by the model. The corresponding spectral-gap estimate is:

$$\hat{\gamma}_{g,c} = s_{g,c}\sigma_\gamma + \mu_\gamma.$$

The first objective component is the pointwise regression loss:

$$\mathcal{L}_{\text{reg}} = \frac{1}{GK}\sum_{g=1}^{G}\sum_{c=1}^{K}\text{SmoothL1}\left(s_{g,c} - t_{g,c}\right).$$

Because candidate selection depends on score differences within the same context, a centered regression component is also included. The group means are:

$$\bar{s}_g = \frac{1}{K}\sum_{c=1}^{K} s_{g,c},$$

$$\bar{t}_g = \frac{1}{K}\sum_{c=1}^{K} t_{g,c},$$

and the centered loss is:

$$\mathcal{L}_{\text{center}} = \frac{1}{GK}\sum_{g=1}^{G}\sum_{c=1}^{K}\text{SmoothL1}\left(\left(s_{g,c} - \bar{s}_g\right) - \left(t_{g,c} - \bar{t}_g\right)\right).$$

A listwise ranking component encourages the model to assign more probability to candidates with larger exact spectral gaps. The target and predicted candidate distributions are defined as:

$$\pi_{g,c} = \frac{\exp\left(\left(\gamma_{g,c} - \bar{\gamma}_g\right)/\tau\right)}{\sum_{j=1}^{K}\exp\left(\left(\gamma_{g,j} - \bar{\gamma}_g\right)/\tau\right)},$$

$$\hat{\pi}_{g,c} = \frac{\exp\left(\left(\hat{\gamma}_{g,c} - \bar{\hat{\gamma}}_g\right)/\tau\right)}{\sum_{j=1}^{K}\exp\left(\left(\hat{\gamma}_{g,j} - \bar{\hat{\gamma}}_g\right)/\tau\right)},$$

where $\tau = 0.002$.

The listwise loss is:

$$\mathcal{L}_{\text{rank}} = -\frac{1}{G}\sum_{g=1}^{G}\sum_{c=1}^{K}\pi_{g,c}\log\left(\hat{\pi}_{g,c}\right).$$

The complete training objective is:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{reg}} + \mathcal{L}_{\text{center}} + 0.1\mathcal{L}_{\text{rank}}.$$

Optimization uses AdamW with learning rate 10^{-3}, weight decay 10^{-4}, and dropout probability 0.1. Training proceeds for at most 30 epochs with early-stopping patience 5. The number of context groups per minibatch is 32, 16, and 8 for $r = 8$, $r = 16$, and $r = 32$, respectively. Each context group contains all 32 candidates. The checkpoint with the smallest validation loss is retained. Independent runs use seeds 20260723, 20260724, and 20260725.

Regression quality is evaluated using mean absolute error, root-mean-square error, and Pearson correlation between predicted and exact gaps. Candidate ordering is evaluated separately using within-context Spearman correlation, top-1 recovery, top-4 recovery, selected-candidate percentile, and spectral regret. For context g, spectral regret is defined as:

$$\text{Regret}_g = \max_{c \in C_r}\gamma_{g,c} - \gamma_{g,\hat{c}},$$

where:

$$\hat{c} = \underset{c \in C_r}{\text{argmax}}\,\hat{\gamma}_{g,c}.$$

The uplift over expected random selection is:

$$\text{Uplift}_g = \gamma_{g,\hat{c}} - \frac{1}{K}\sum_{c=1}^{K}\gamma_{g,c}.$$

These selection metrics distinguish a model that merely predicts the average gap of a context from a model that can rank candidate increment tables within that context.

VI. INTEGRATION INTO POLLARD'S RHO WITH r-ADDING WALK

The integrated solver receives a short Weierstrass elliptic curve:

$$E: y^2 = x^3 + ax + b \pmod p$$

a cyclic subgroup of prime order N, a generator P, and a target point:

$$Q = kP,$$

where $k \in \mathbb{Z}_N$ is the unknown discrete logarithm. The neural configuration module is invoked once before the iterative phase of Pollard's Rho method. It produces a table:

$$\mathcal{M}_r = \left\{\left(\alpha_i, \beta_i, M_i\right)\right\}_{i=0}^{r-1},$$

where:

$$M_i = \alpha_i P + \beta_i Q.$$

The proposed framework supports two configuration modes. In the direct-generation mode, NM-r first predicts:

$$\hat{r} \in \{8, 16, 32, 64\},$$

after which NM-inc predicts the normalized coefficient table conditioned on $\hat{r}$. The predictions are decoded and postprocessed according to the deterministic procedure described in Section IV.

In the candidate-scoring mode, a value:

$$\hat{r} \in \{8, 16, 32\}$$

is fixed for the experiment. A pseudorandom generator with a recorded seed constructs a pool of K valid coefficient tables:

$$C_r = \left\{C_r^{(1)}, \ldots, C_r^{(K)}\right\}.$$

The trained candidate-scoring model assigns a predicted spectral gap to every table and selects:

$$\hat{c} = \underset{c \in \{1, \ldots, K\}}{\operatorname{argmax}} \ \hat{\gamma}_\theta\left(C_r^{(c)} \mid p, a, b, N, P, Q\right).$$

The selected coefficient table $C_r^{(\hat{c})}$ is then used to construct the increment points. This mode replaces an expensive online spectral evaluation with neural inference and comparison of the predicted candidate scores.

After either configuration procedure has produced $\mathcal{M}_r$, the neural module is no longer invoked during the walk. The selected value of r, coefficient pairs, increment points, and tagging rule remain fixed for the duration of the run. The subsequent state updates, collision detection, discrete-logarithm recovery, and restart handling are performed by the classical algorithmic core of Pollard's Rho method.

The iterative phase begins by sampling two independent scalars $u_0, v_0 \xleftarrow{S} \mathbb{Z}_N$ and constructing the initial state:

$$X_0 = u_0 P + v_0 Q.$$

Every state is stored together with its coefficient representation:

$$S_t = \left(X_t, u_t, v_t\right),$$

which satisfies the invariant:

$$X_t = u_t P + v_t Q.$$

The deterministic tagging function maps an elliptic-curve point to one of the r increment-table indices. For an affine point $X = \left(x_X, y_X\right)$, it is defined as:

$$h_r(X) = x_X \bmod r.$$

Because all supported values of r are powers of two, this operation is equivalent to selecting the least significant $\log_2(r)$ bits of the canonical integer representation of x_X. The point at infinity is assigned the fixed tag:

$$h_r(\mathcal{O}) = 0.$$

At iteration t, the current table index is:

$$i_t = h_r\left(X_t\right).$$

The corresponding increment entry is:

$$\left(\alpha_{i_t}, \beta_{i_t}, M_{i_t}\right),$$

where:

$$M_{i_t} = \alpha_{i_t} P + \beta_{i_t} Q.$$

One transition of the r-adding walk is then calculated as:

$$\begin{aligned} X_{t+1} &= X_t + M_{i_t}, \\ u_{t+1} &= \left(u_t + \alpha_{i_t}\right) \bmod N, \\ v_{t+1} &= \left(v_t + \beta_{i_t}\right) \bmod N. \end{aligned}$$

These updates preserve the invariant because:

$$\begin{aligned} X_{t+1} &= X_t + M_i \\ &= u_t P + v_t Q + \alpha_{i_t} P + \beta_{i_t} Q \\ &= \left(u_t + \alpha_{i_t}\right) P + \left(v_t + \beta_{i_t}\right) Q \\ &= u_{t+1} P + v_{t+1} Q. \end{aligned}$$

Thus, the neural module changes only the configuration of the r-adding walk. It does not modify the algebraic state invariant or the collision-recovery mechanism of Pollard's Rho method.

Collision detection uses Floyd's cycle-finding procedure. The slow state applies one r-adding transition during each outer iteration, whereas the fast state applies two transitions. Let:

$$S_s = \left(X_s, u_s, v_s\right),$$

and:

$$S_f = \left(X_f, u_f, v_f\right),$$

denote the slow and fast states. A collision is detected when:

$$X_s = X_f.$$

By the maintained coefficient invariant:

$$X_s = u_s P + v_s Q,$$

and:

$$X_f = u_f P + v_f Q.$$

Using $Q = kP$, equality of the two states implies:

$$u_s P + v_s kP = u_f P + v_f kP.$$

Therefore:

$$\left(u_s - u_f\right) P = \left(v_f - v_s\right) kP.$$

Because P generates a subgroup of prime order N, the candidate discrete logarithm is recovered as:

$$\hat{k} = \left(u_s - u_f\right)\left(v_f - v_s\right)^{-1} \bmod N,$$

provided that:

$$v_f - v_s \not\equiv 0 \pmod{N}.$$

The recovered value is verified by checking:

$$\hat{k}P = Q.$$

If the denominator is zero modulo N, the collision is degenerate and does not determine k. A failed verification is treated in the same manner. In either case, new initial coefficients are sampled:

$$u'_0 v'_0 \xleftarrow{S} \mathbb{Z}_N,$$

and the walk is restarted from:

$$X'_0 = u'_0 P + v'_0 Q.$$

The selected configuration $\mathcal{M}_r$ remains unchanged across such restarts. Consequently, the neural module controls the increment table, whereas the initialization randomness and collision processing remain part of the classical Pollard's Rho method.

VII. EXPERIMENTAL RESULTS

A. Experimental protocol. The experiments comprise two stages. The first stage evaluates direct prediction with NM-r and NM-inc using the approximate-gap dataset. The second stage evaluates neural candidate scoring using the exact-gap dataset and measures the effect of the selected increment tables on Pollard's Rho method.

The candidate-scoring models are trained separately for $r = \{8, 16, 32\}$. Three independently initialized models are retained for each r, using seeds 20260723, 20260724, and 20260725. Their predicted spectral gaps are averaged during candidate selection:

$$\hat{\gamma}^{(c)}_{\text{ens}} = \frac{1}{3}\sum_{s=1}^{3} \hat{\gamma}^{(c)}_s.$$

Based on the representation comparison, the frozen downstream selection policy uses the point representation for $r = 8$, the sampled $L = 128$ transition-matrix representation for $r = 16$, and the coefficient representation for $r = 32$. The selected representations and model checkpoints are fixed before executing the Pollard benchmark.

The benchmark uses 500 held-out curve contexts from the test subset of the exact-gap dataset. Their prime subgroup orders satisfy $2^{15} \le N < 2^{20}$.

For every context and every r, the same pool of $K = 32$ candidate increment tables is used to compare four selection policies:

- Random: one candidate is selected uniformly from the pool.
- Neural: the candidate having the largest ensemble-predicted spectral gap is selected.
- Sampled oracle: the candidate having the largest spectral gap estimated with $L = 512$ observations per transition-matrix row is selected.
- Exact oracle: the candidate having the largest exact reduced-matrix spectral gap is selected.

The sampled and exact oracles are evaluation references rather than neural networks. The exact oracle identifies the best candidate according to the investigated spectral criterion within the finite pool; it is not assumed to be an oracle for the minimum execution time of Pollard's Rho method.

For each combination of context, r, and selection policy, 30 trials are executed. The initialization randomness is paired across the four policies within each trial. The complete benchmark therefore contains $500 \times 3 \times 4 \times 30 = 180000$ measurements. All 180000 runs recovered the correct discrete logarithm.

The primary outcome is the number of r-adding transitions required to recover the discrete logarithm. To compare contexts having different subgroup orders, the iteration count is normalized as:

$$I_{\text{norm}} = \frac{I}{\sqrt{N}}.$$

For each context and policy, normalized iterations are first averaged across the 30 paired trials. Statistical resampling then uses the curve context, rather than an individual trial, as the independent unit. Confidence intervals are calculated using 50000 bootstrap samples, and two-sided paired sign-flip tests use 100000 permutations. Holm correction is applied both to the three planned neural-versus-random comparisons and to the family of nine per-r comparisons.

Dataset manifests, generation seeds, split identifiers, model checkpoints, raw Pollard measurements, configuration files, tables, figures, and SHA-256 hashes are retained as reproducibility artifacts. Neural candidate selections are generated and frozen before the C# benchmark. Consequently, the reported iteration and walk-time comparisons exclude neural-inference latency and measure the behavior of Pollard's Rho method after configuration selection.

B. Direct-prediction results. The approximate-gap dataset produced a strongly imbalanced distribution of selected table sizes. Among the 5000 context targets, 4072 selected $r = 8$, 801 selected $r = 16$, 117 selected $r = 32$, and 10 selected $r = 64$. The corresponding proportions were 81.44%, 16.02%, 2.34%, and 0.20%, respectively.

The frozen test subset contained 500 contexts. Its target distribution was 416 instances of $r = 8$, 73 instances of $4 = 16$, 11 instances of $r = 32$, and no instance of $r = 64$. Table 1 summarizes the NM-r results.

TABLE 1. NM-r classification results.

Model	Accuracy	Macro-F1	Prediction behavior
Majority-class baseline	0.832	0.227	Predicts $r = 8$ for all contexts
Unweighted NM-r	0.832	0.227	Predicts $r = 8$ for all contexts
Class-weighted NM-r	0.148	0.065	Predicts $r = 16$ for 499 of 500 contexts

Although the unweighted NM-r model achieved an accuracy of 83.2%, it exactly reproduced the majority-class baseline and did not correctly separate the minority classes. The weighted model changed the dominant predicted class but reduced accuracy to 14.8% and macro-F1 to 0.065. Therefore, neither loss formulation demonstrated context-

dependent prediction of r.

The stability of the approximate spectral labels was examined by reevaluating the same 800 candidate tables from 50 test contexts using $L_{\text{main}} = 512$ and $L_{\text{HP}} = 4096$.

The Pearson correlation between the two sets of candidate-gap estimates was:

$$\varrho_{\text{Pearson}} = 0.0202\varrho.$$

The same candidate table was selected at both sampling levels in only 5 of 50 contexts, giving a target-agreement rate of 0.1.

Thus, increasing L by a factor of eight frequently changed which particular coefficient table had the largest estimated spectral gap. This result indicates that the single coefficient vector selected at $L = 512$ was not a sufficiently stable regression target.

The NM-inc architecture was implemented, but its one-to-one target formulation was not promoted to the downstream Pollard benchmark after this stability check. A coefficient-wise mean absolute error against an unstable selected table would primarily measure reproduction of a sampling-dependent target and would not establish spectral quality. Accordingly, the direct NM-inc output was not used to claim successful increment generation.

These findings do not show that neural configuration of the r-adding walk is impossible. They show that direct classification of r and one-to-one regression of a single sampled winner were not supported by the approximate-gap dataset. This negative result motivated the candidate-scoring formulation, in which all candidates receive continuous exact-gap targets and the model learns spectral-quality estimation and within-context ranking instead of reproducing one arbitrarily selected coefficient table.

C. Exact-gap prediction and candidate ranking. Table 2 reports the held-out results for the candidate representations retained for the downstream experiment and for the matrix-aware attention model. The displayed single-seed results use seed 20260723.

TABLE 2. Exact-gap prediction and within-context ranking.

r	Representation	MAE	Pearson	Within-group Spearman	Top-1	Uplift
8	Points	0.001727	0.3695	0.0051	0.014	0.000157
8	Attention, $L = 512$	0.001602	0.9783	–0.0131	—	—
16	Matrix, $L = 128$	0.001555	0.4576	0.0101	0.024	0.000187
16	Attention, $L = 512$	0.001420	0.9867	–0.0109	—	—
32	Coefficients	0.001847	0.3355	–0.0090	0.016	0.000558
32	Attention, $L = 512$	0.001300	0.9971	–0.0089	—	—

The matrix-aware model achieved Pearson correlations between 0.9783 and 0.9971 when candidates from all test contexts were pooled. However, its mean within-context Spearman correlations remained close to zero. The context-only baseline similarly produced nonzero pooled Pearson correlations of 0.3651, 0.4564, and 0.3358 for $r = 8$, $r = 16$ and $r = 32$, respectively, despite containing no candidate-specific information.

This ablation demonstrates that a large pooled correlation does not necessarily imply useful candidate ranking. A model can estimate the average spectral-gap level of a curve context while failing to distinguish among the 32 increment tables belonging to that context. Within-context correlation, spectral uplift, and selection regret are therefore more relevant than pooled regression correlation for the intended selection task.

Across the three training seeds, the point representation for $r = 8$ produced mean uplift 7.4×10^{-5} with mean regret 0.003428. The $L = 128$ matrix representation for $r = 16$ produced mean uplift 1.00×10^{-4} with mean regret 0.002971. The coefficient representation for $r = 32$ produced mean uplift -8.9×10^{-5} with mean regret 0.003279. The seed-level results were variable, and the uplift confidence intervals generally included zero.

The top-1 recovery rates were also low. For the retained seed-20260723 models, they were 1.4%, 2.4%, and 1.6% for $r = 8$, $r = 16$, and $r = 32$, respectively. For comparison, uniform selection from 32 candidates has an expected top-1 probability of $1/32 = 3.125\%$. Consequently, the candidate scorer did not reliably recover the exact spectral-gap oracle.

Memorization experiments were conducted to distinguish insufficient model capacity from limited generalization. When trained and evaluated on the same 100 context groups, the retained architectures achieved within-group Spearman correlations of 0.9719, 0.8358, and 0.9563 for $r = 8$, $r = 16$, and $r = 32$, respectively. Their training-set top-1 recovery rates were 83%, 58%, and 71%. The models could therefore fit finite candidate groups, but the learned ordering did not transfer reliably to unseen curve contexts.

Training-set fractions of 10%, 25%, 50%, and 100% were also compared. Increasing the number of training contexts did not produce a monotonic improvement in held-out uplift or within-context Spearman correlation. Within the investigated dataset size and representations, the primary limitation was therefore not resolved solely by adding more training groups.

Despite weak exact-oracle recovery, candidate-scoring checkpoints were retained for the downstream empirical test. This test addresses a different question: whether increment tables selected by the frozen neural policy affect the measured iteration count of Pollard's Rho method, even when the model does not consistently identify the maximum-gap candidate.

D. Pollard iteration results. All four selection policies successfully recovered the discrete logarithm in every trial.

The overall success rate was therefore:

$$\frac{18000}{18000} = 100\%.$$

No restart was required in the recorded benchmark runs. Table 3 presents the mean normalized iteration counts. The same comparison is visualized in Fig. 1, which shows the mean normalized iteration count and its uncertainty for each selection policy and value of r.

TABLE 3. Mean normalized iterations of Pollard's Rho method.

r	Random	Neural	Sampled oracle	Exact oracle
8	1.3954	1.3222	1.3421	1.3524
16	1.3092	1.2823	1.3040	1.3174
32	1.2933	1.2927	1.2890	1.2631

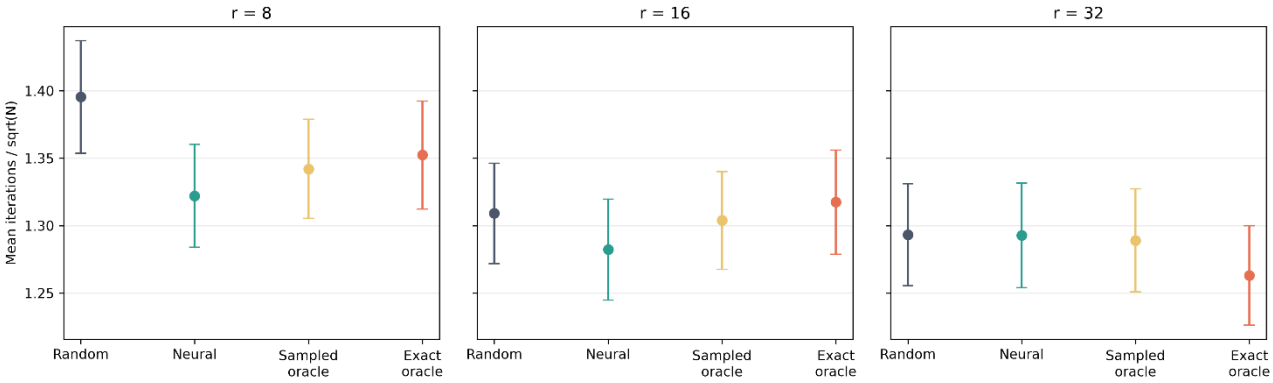

FIG. 1. Mean iteration count of Pollard's Rho method normalized by $\sqrt{N}$ for random, neural, sampled-oracle, and exact-oracle candidate selection. Points represent context-level means; error bars show 95% normal-approximation confidence intervals.

For $r = 8$, the neural policy reduced the mean normalized iteration count by 0.07325, which corresponds to a relative reduction of 5.25%. The context-level bootstrap interval for the relative reduction was

$$95\%\ \mathrm{CI} = [1.40\%, 9.00\%].$$

The paired sign-flip test produced $p = 0.0079$. After Holm correction across the three planned neural-versus-random comparisons, the adjusted value was $p_{\mathrm{Holm},3} = 0.0237$. However, after correction across all nine exploratory per-r comparisons, it became $p_{\mathrm{Holm},9} = 0.0711$.

Thus, the $r = 8$ result remained significant within the planned neural comparison family but not within the broader nine-comparison family.

For $r = 16$, the estimated relative reduction was 2.06%, with:

$$95\%\ \mathrm{CI} = [-1.83\%, 5.81\%],$$

and:

$$p = 0.9829.$$

The experiments therefore did not establish a conclusive neural advantage for $r = 16$ or $r = 32$.

When the three values of r were aggregated at the context level, the neural policy reduced normalized iterations by 0.03359 corresponding to a relative reduction of 2.52%. The aggregate bootstrap interval was:

$$95\%\ \mathrm{CI} = [0.24\%, 4.79\%],$$

and the paired sign-flip test produced $p = 0.03248$.

Table 4 summarizes the neural-versus-random effects.

TABLE 4. Paired neural-versus-random comparisons.

Scope	Relative reduction	95% CI	Context win rate	p- value
$r = 8$	5.25%	[1.40%, 9.00%]	54.0%	0.0079
$r = 16$	2.06%	[−1.83%, 5.81%]	52.4%	0.2995
$r = 32$	0.05%	[−4.15%, 4.04%]	49.8%	0.9829
All r	2.52%	[0.24%, 4.79%]	51.2%	0.03248

The sampled oracle produced an aggregate relative reduction of 1.57%, with a confidence interval of [−0.88%, 3.97%]. The exact oracle produced an aggregate relative reduction of 1.63%, with:

$$95\%\ \mathrm{CI} = [-0.82\%, 3.98\%].$$

Neither oracle comparison was statistically conclusive. Therefore, selecting the largest estimated or exact reduced-matrix spectral gap within a pool of 32 candidates did not consistently minimize the iteration count of Pollard's Rho method. The per-r effect estimates and their confidence intervals are summarized in Fig. 2.

Within the investigated toy-field regime, the frozen neural policy nevertheless demonstrated a modest aggregate empirical advantage over random candidate selection. The strongest individual result occurred for $r = 8$; the evidence does not support a universal improvement across all examined values of r.

E. Spectral-gap Association and execution time. To evaluate the spectral gap as a proxy for practical walk behavior, the change in exact spectral gap relative to

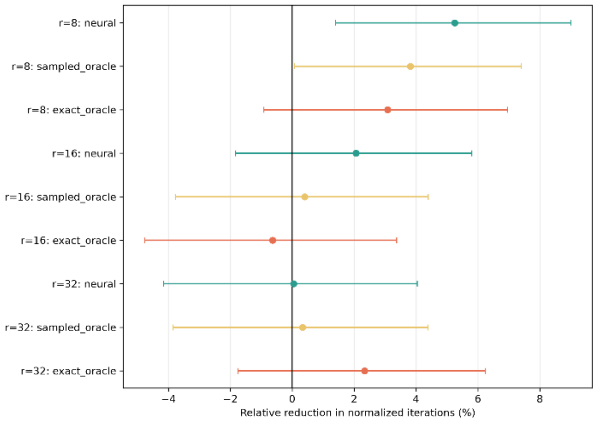

FIG. 2. Relative reduction in the normalized iteration count of Pollard's Rho method compared with random candidate selection. Positive values indicate fewer iterations than random selection; error bars show context-level 95% bootstrap confidence intervals.

random selection was compared with the corresponding reduction in normalized iterations. Table 5 presents context-level Pearson and Spearman correlations.

All observed correlations were weak. In particular, the Spearman correlations for the exact-oracle comparisons ranged from 0.0010 to 0.0590. Thus, contexts in which the exact oracle produced a larger spectral-gap increase were not consistently the contexts in which it produced a larger iteration reduction. The corresponding context-level relationship for neural versus random candidate selection is visualized in Fig. 3.

The result does not imply that the spectral properties of the walk are irrelevant. It shows that the exact spectral gap of the reduced $r \times r$ tag-transition matrix was not, by itself, a strong context-level predictor of finite-run collision time in the investigated regime. The reduced matrix describes tag mixing, whereas the collision time also depends on the algebraic state trajectory, increment interactions, initialization, and finite-sample variability.

TABLE 5. Association between spectral-gap change and iteration reduction.

r	Comparison	Pearson	Spearman
8	Neural vs. random	–0.0406	–0.0399
8	Sampled oracle vs. random	–0.0149	–0.0466
8	Exact oracle vs. random	0.0454	0.0368
16	Neural vs. random	–0.0772	–0.0476
16	Sampled oracle vs. random	0.0343	0.0208
16	Exact oracle vs. random	0.0035	0.0010
32	Neural vs. random	–0.1299	0.0281
32	Sampled oracle vs. random	–0.0986	0.0640
32	Exact oracle vs. random	–0.1023	0.0590

The selection policies generally returned different tables. The neural and exact-oracle policies selected the same candidate in 2.4%, 3.4%, and 3.6% of the contexts for $r = 8$, $r = 16$, and $r = 32$ respectively. The sampled and exact oracles agreed in 2.6%, 4.2%, and 3.6% of the contexts. These rates are close to the uniform finite-pool reference:

$$\frac{1}{32} = 3.125\%.$$

Therefore, the neural, sampled-oracle, and exact-oracle policies should not be interpreted as approximately equivalent selectors.

A separate single-worker timing check was performed on 50 held-out contexts with 10 trials per context and r. The mean walk times were below 0.56 ms for all methods. At this scale, elapsed-time measurements were sensitive to runtime overhead, scheduling, and timer resolution. For example, the neural policy reduced the iteration count for $r = 8$ but did not reduce the corresponding mean single-worker wall-clock time.

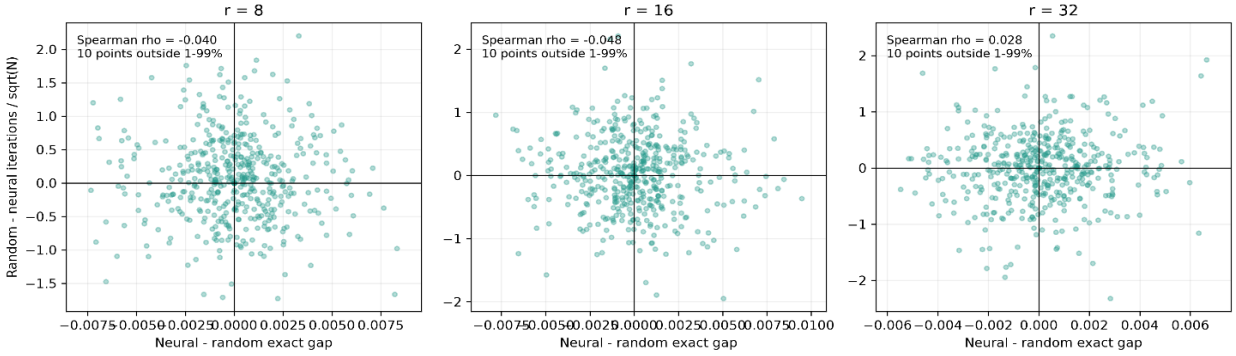

FIG. 3. Context-level relationship between the exact spectral-gap difference of neural and random selections and the corresponding reduction in normalized iterations. Positive vertical-axis values indicate fewer iterations for the neural policy. Spearman coefficients are calculated using all 500 contexts for each r; points outside the 1%-99% horizontal-axis quantile range are omitted from the visualization only.

For this reason, normalized iteration count was selected as the primary implementation-independent outcome. Wall-clock measurements are retained as supplementary descriptive results. Neural-inference latency is excluded because all candidate selections were frozen before the C# benchmark. Consequently, the reported timing results characterize the configured walk and must not be interpreted as end-to-end neural-module latency.

VIII. DISCUSSION AND LIMITATIONS

A. Interpretation of the two neural approaches. The experiments distinguish two formulations of neural r-adding walk configuration. The direct NM-r + NM-inc formulation assumes that every curve context has a sufficiently stable single target: one value of r and one coefficient table. The results did not support this

assumption. NM-r reproduced the majority-class baseline, whereas high-precision reevaluation changed the selected coefficient table in 90% of the examined contexts.

This instability does not imply that all candidate tables are equally effective or that neural configuration is intrinsically infeasible. It indicates that selecting one winner from a small candidate pool using a sampled spectral estimate produces a noisy one-to-one supervision target. Multiple tables may have close estimated gaps, and their ordering may change when the transition sample is enlarged.

The candidate-scoring formulation avoids coefficient-wise reproduction of a single winner. It assigns a continuous spectral target to every candidate and allows the model to compare independently generated tables. This formulation is therefore methodologically better aligned with the available data. Nevertheless, its held-out within-context ranking performance remained weak, even when pooled exact-gap regression correlations were high.

The Pollard benchmark produced a different and practically relevant result. The frozen neural policy reduced normalized iterations by 2.52% in aggregate and by 5.25% for $r = 8$. However, the model did not reliably recover the exact spectral-gap oracle, and exact-gap improvements were only weakly associated with iteration reductions. The empirical improvement therefore cannot be attributed solely to consistent maximization of the reduced-matrix spectral gap.

A cautious interpretation is that the neural policy captured some candidate or context characteristics that were useful for the tested walks, while the investigated scalar spectral criterion provided only a partial description of finite-run collision behavior. Replication on independently generated datasets is required before treating the observed reduction as a general effect.

B. Limitations. First, the experiments use controlled research-scale groups. The approximate-gap dataset covers $2^{25} \le p < 2^{30}$, whereas exact enumeration and the Pollard benchmark use $2^{15} \le p, N < 2^{20}$. The results must not be extrapolated directly to cryptographic-size elliptic-curve groups.

Second, the exact spectral gap is exact only for the reduced $r \times r$ tag-transition matrix. It is not the spectral gap of a complete $N \times N$ state-transition matrix. Reduction to tag dynamics discards information about individual group states and algebraic dependencies between successive increments.

Third, the exact-gap dataset and Pollard benchmark cover $r \in \{8,16,32\}$, but not $r = 64$. The direct dataset includes $r = 64$, although this class accounts for only 0.20% of its targets and is therefore insufficiently represented for reliable classification.

Fourth, every candidate-selection policy is restricted to a pool of 32 randomly generated tables. The exact oracle is consequently a best-of-32 oracle rather than a global optimum over all possible coefficient tables. A different pool size or generation distribution may change both spectral and Pollard results.

Fifth, candidate representations have different online costs. Coefficient and increment-point features can be calculated without constructing a transition matrix, whereas the $L = 128$ and $L = 512$ representations require sampled transition observations. The present benchmark freezes candidate selections before execution and excludes neural-inference latency. It therefore measures walk quality rather than complete end-to-end configuration cost.

Sixth, the representation comparison and the selection of the downstream per-r models were exploratory. Although Pollard trials used fresh paired initialization randomness, confirmation on a separately generated external context set is required to eliminate selection-related optimism.

Finally, the strongest individual neural result was obtained for $r = 8$. The confidence intervals for $r = 16$ and $r = 32$ included zero, and the $r = 8$ result did not remain significant after Holm correction across all nine exploratory comparisons. The evidence therefore supports a limited empirical improvement in the investigated setting rather than a universal superiority claim.

C. Directions for further research. Future dataset generation should replace single-winner labels with uncertainty-aware targets. Possible alternatives include elite candidate sets, soft target distributions, repeated spectral measurements, and adaptive selection of L until the confidence intervals of leading candidates can be separated.

Candidate models should also be designed to respect the set structure of increment tables. Permutation-invariant set networks, attention over increment points, and graph-based representations of transitions may generalize more effectively than flattening coefficient vectors.

The spectral target can be extended from one scalar to a vector of transition characteristics, including several leading eigenvalue magnitudes, row imbalance, entropy, and uncertainty of the sampled matrix. Such multi-characteristic supervision may retain information that is lost when every candidate is represented only by:

$$\gamma(P) = 1 - \max_{2 \le k \le r} |\lambda_k(P)|.$$

Further experiments should separate model-development contexts from a final untouched replication set, evaluate larger candidate pools, include $r = 64$, and measure the complete configuration latency. Validation on Pollard's Rho method should remain an empirical evaluation stage even when the neural model is trained exclusively using spectral targets.

IX. CONCLUSION

This study developed and experimentally evaluated a reproducible framework for neural configuration of the r-adding walk in Pollard's Rho method for the elliptic curve discrete logarithm problem. Two related approaches were investigated: direct generation of walk parameters through NM-r and NM-inc, and neural scoring of independently generated candidate increment tables.

The direct approach used an approximate-gap dataset containing 5000 curve contexts and 80000 candidate configurations for $r \in \{8,16,32,64\}$.

NM-r achieved 83.2% test accuracy, but this result was identical to the majority-class baseline because the model predicted $r = 8$ for every test context. High-precision reevaluation increased the transition sample from $L = 512$ to $L = 4096$, but the same coefficient table remained selected in only 10% of the examined contexts. Consequently, direct one-to-one prediction of a single selected table was not

supported by sufficiently stable labels.

The second approach reformulated neural configuration as exact-gap regression and within-context candidate ranking. Its dataset contained 5000 contexts and 480000 candidate labels for $r \in \{8,16,32\}$.

Although the trained models did not reliably recover the exact best-of-32 spectral candidate, their frozen selections were evaluated in a paired C# implementation of Pollard's Rho method. The benchmark contained 180000 successful discrete-logarithm measurements over 500 held-out contexts.

Across $r = 8$, $r = 16$, and $r = 32$, neural selection reduced the normalized iteration count by 2.52% relative to random selection, with 95% CI = [0.24%,4.79%] and $p = 0.03248$. For $r = 8$, the reduction was 5.25%, with 95% CI = [1.40%,9.00%]. This result remained significant after Holm correction across the three planned neural comparisons, with $p_{\text{Holm},3} = 0.0237$, but not after correction across all nine exploratory per-r comparisons.

The exact and sampled spectral-gap oracles did not consistently reduce Pollard iterations, and the relationship between spectral-gap improvement and iteration reduction was weak. Therefore, the reduced tag-transition spectral gap should be treated as an informative but incomplete proxy for finite-run collision behavior rather than as a guaranteed optimizer of Pollard's Rho method.

The obtained results support the feasibility of neural candidate selection as an experimental configuration mechanism in the investigated toy-field regime, while direct single-target parameter generation remains unresolved. Future work should introduce uncertainty-aware or elite-set supervision, permutation-invariant candidate representations, larger independent replication datasets, complete configuration-latency measurements, and experiments on larger subgroup orders.

AUTHOR CONTRIBUTIONS

M.O., D.H. – conceptualization, problem formulation, literature review, analysis of discrete logarithm methods, development of the research methodology, analysis of the results obtained, articulation of conclusions, review, and editing.

COMPETING INTERESTS

The authors declare no competing interests.

REFERENCES

[1] V. S. Miller, "Use of elliptic curves in cryptography," in *Advances in Cryptology – CRYPTO '85*, Lecture Notes in Computer Science, vol. 218. Berlin, Germany: Springer, 1986, pp. 417–426, doi: 10.1007/3-540-39799-X_31.

[2] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, no. 177, pp. 203–209, 1987, doi: 10.1090/S0025-5718-1987-0866109-5.

[3] J. M. Pollard, "Monte Carlo methods for index computation (mod p)," *Mathematics of Computation*, vol. 32, no. 143, pp. 918–924, 1978, doi: 10.1090/S0025-5718-1978-0491431-9.

[4] E. Teske, "Speeding up Pollard's rho method for computing discrete logarithms," in *Algorithmic Number Theory*, Proc. ANTS-III, Lecture Notes in Computer Science, vol. 1423. Berlin, Germany: Springer, 1998, pp. 541–554, doi: 10.1007/BFb0054891.

[5] E. Teske, "On random walks for Pollard's rho method," *Mathematics of Computation*, vol. 70, no. 234, pp. 809–825, 2001, doi: 10.1090/S0025-5718-00-01213-8.

[6] J. Jebrane, A. Chhaybi, S. Lazaar, and A. Nitaj, "Elliptic curve cryptography with machine learning," *Cryptography*, vol. 9, no. 1, Art. no. 3, 2025, doi: 10.3390/cryptography9010003.

[7] D. Hankerson, A. J. Menezes, and S. A. Vanstone, *Guide to Elliptic Curve Cryptography*. New York, NY, USA: Springer, 2004, doi: 10.1007/b97644.

[8] S. D. Miller and R. Venkatesan, "Spectral analysis of Pollard Rho collisions," in *Algorithmic Number Theory—ANTS VII*, Lecture Notes in Computer Science, vol. 4076. Berlin, Germany: Springer, 2006, pp. 573–581, doi: 10.1007/11792086_40.

[9] M. Al-Khalidi, R. Al-Zaidi, T. Ali, S. Khan, and A. K. Bashir, "AI-optimized elliptic curve with certificate-less digital signature for zero trust maritime security," *Ad Hoc Networks*, vol. 166, Art. no. 103669, 2025, doi: 10.1016/j.adhoc.2024.103669.

[10] K. Javeed, A. El-Moursy, and D. Gregg, "EC-Crypto: Highly efficient area-delay optimized elliptic curve cryptography processor," *IEEE Access*, vol. 11, pp. 56649–56662, 2023, doi: 10.1109/ACCESS.2023.3282781.

[11] R. Ifrim, D. Loghin, and D. Popescu, "A systematic review of fast, scalable, and efficient hardware implementations of elliptic curve cryptography for blockchain," *ACM Transactions on Reconfigurable Technology and Systems*, vol. 17, no. 4, Art. no. 62, pp. 1–33, 2024, doi: 10.1145/3696422.

[12] D. Maimuț and A. C. Matei, "Speeding-up elliptic curve cryptography algorithms," *Mathematics*, vol. 10, no. 19, Art. no. 3676, 2022, doi: 10.3390/math10193676.

[13] D. A. Levin and Y. Peres, *Markov Chains and Mixing Times*, 2nd ed. Providence, RI, USA: American Mathematical Society, 2017, doi: 10.1090/mbk/107.

Mykola Onai

PhD in Computer Systems and Components, Associate Professor at the Computer Systems Software Department, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute". His research interests include applied cryptography, elliptic curve cryptography, discrete logarithm problem, secure software engineering, hardware algorithms for cryptography and algorithmic optimization of cryptographic methods. Author of more than 100 scientific publications and 4 patents.

ORCID ID: 0000-0002-4938-8355

Danylo Hulko

Is a Ph.D. student at the Department of Computer Systems Software, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine. His research interests include elliptic curve cryptography, Pollard's Rho method and random-walk-based cryptanalysis, machine learning for cryptographic parameter selection, and the design of reproducible software frameworks for security research.

ORCID ID: 0009-0008-6810-737X

Нейромережеве налаштування r-додавального блукання в ρ-методі Полларда з використанням спектрального розриву для задачі дискретного логарифмування на еліптичних кривих

Микола Онай*, Данило Гулько

Кафедра програмного забезпечення комп'ютерних систем, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна

*Автор-кореспондент (Електронна адреса: onay@pzks.fpm.kpi.ua)

АНОТАЦІЯ У статті досліджено два нейромережеві підходи до конфігурування r-додавального блукання в ρ-методі Полларда для розв'язання задачі дискретного логарифмування на еліптичних кривих у короткій формі Вейєрштрасса над простими полями. У першому підході класифікаційна нейронна мережа NM-r прогнозує розмір r таблиці інкрементів, а регресійна нейронна мережа NM-inc безпосередньо генерує нормалізовані коефіцієнти інкрементів (α_i, β_i) на основі контексту кривої та підгрупи. Для цього підходу було сформовано набір даних із наближеними значеннями спектрального розриву та груповим за контекстами розподілом на навчальну, валідаційну й тестову частини. Оцінювання підходу виявило значний дисбаланс цільових класів для NM-r і нестабільність однозначних цільових таблиць інкрементів за точнішого вибіркового оцінювання матриць переходів. Це обмежило можливість надійного наскрізного узагальнення початкової моделі. Виявлені обмеження мотивували розроблення другого підходу, у якому нейронна мережа оцінює скінченну множину таблиць інкрементів-кандидатів і для фіксованого r обирає кандидата з найбільшою прогнозованою спектральною якістю. Для оцінювання цього підходу без вибіркової невизначеності цільових значень було сформовано точний набір даних, що містить 5000 контекстів еліптичних кривих і 480000 конфігурацій-кандидатів для $r \in \{8, 16, 32\}$. Простий модуль поля p і порядок підгрупи N задовольняли умову $2^{15} < p, N < 2^{20}$. Цільове значення для кожного кандидата обчислювали за точною матрицею переходів між тегами розміру $r \times r$, побудованою шляхом перебору всіх станів відповідної підгрупи. Моделі оцінювання кандидатів навчали з груповим за контекстами розподілом 80/10/10, порівнювали для декількох представлень вхідних даних і випадкових початкових значень, а їхні параметри фіксували до інтеграції з реалізацією ρ-методу Полларда мовою C#. Під час експериментального порівняння досліджували нейромережевий вибір, детермінований випадковий вибір, оракул наближеного спектрального розриву з $L = 512$ та оракул точного спектрального розриву. Експеримент виконано на 500 відкладених тестових контекстах із 30 парними запусками для кожного контексту та значення r, унаслідок чого отримано 180000 успішних вимірювань. Для сукупності $r \in \{8, 16, 32\}$ нейромережевий вибір зменшив середню кількість ітерацій, нормалізовану за $\sqrt{N}$, на 2.52% порівняно з випадковим вибором. Контекстний бутстреп-інтервал із довірчою ймовірністю 95% становив [0.24%, 4.79%], а значення парного рандомізаційного тесту зі зміною знаків становило $p = 0.0325$. Для $r = 8$ зменшення кількості ітерацій становило 5.25%, довірчий інтервал із імовірністю 95% дорівнював [1.40%, 9.00%], а скориговане за методом Холма значення для трьох порівнянь нейромережевого та випадкового вибору становило $p = 0.0237$. Для $r = 16$ і $r = 32$ переконливого покращення не отримано. Крім того, збільшення точного спектрального розриву мало лише слабкий зв'язок зі зменшенням кількості ітерацій ρ-методу Полларда, а оракул точного спектрального розриву не забезпечив сталої переваги. Отримані результати демонструють помірний емпіричний виграш від нейромережевого оцінювання кандидатів у дослідженому експериментальному діапазоні, але водночас показують, що спектральний розрив сам по собі не є універсальним предиктором ефективності ρ-методу Полларда.

КЛЮЧОВІ СЛОВА інженерія програмного забезпечення, задача дискретного логарифмування на еліптичних кривих, ρ-метод Полларда, r-додавальне блукання, нейронні мережі.