Journal of Discrete Mathematical Sciences & Cryptography
ISSN 0972-0529 (Print), ISSN 2169-0065 (Online)
Vol. 28 (2025), No. 5-B, pp. 2071–2080
DOI : 10.47974/JDMSC-2424

AI-augmented cryptanalysis using combinatorics and reinforcement learning

Payal Garg [§]
Department of Computer Science & Engineering
GL Bajaj Institute of Technology and Management
Greater Noida 201306
Uttar Pradesh
India

Aruna Verma [†]
Department of Computer Science and Application
Vivekananda Global University
Jaipur 303012
Rajasthan
India

Sushama Tanwar *
Department of Computer Science & Engineering
Manipal University Jaipur
Jaipur 303007
Rajasthan
India

Pankaj Dadheech [‡]
Department of Computer Science & Engineering
Swami Keshvanand Institute of Technology,
Management & Gramothan
Jaipur 302017
Rajasthan
India

[§] *E-mail:* payalgarg.cs@gmail.com
[†] *E-mail:* vermaaruna1995@gmail.com
* *E-mail:* sushama.tanwar@jaipur.manipal.edu(Corresponding Author)
[‡] *E-mail:* pankajdadheech777@gmail.com

© Taru Publications

Abstract

On one side, the modern day symmetric-key cryptography protects everything from our online banking to critical infrastructures, and the giant key-space of AES-128 alone around 3.4×10^{38} possibilities makes the brute-force attack impractical, which makes sense for the development of more sophisticated search techniques. Although traditional combinatorial attacks eliminate keys based on algebraic relationships, they have exponential complexity even for low-degree equations and are not applicable to multiple cipher designs. At the same time, advances in machine learning that have been made recently demonstrate that reinforcement learning agents can learn to control search heuristics, cutting solution times by as much as a factor of ten in closely associated optimization problems and greatly enhanced success rates in side-channel key recovery. In this paper, we bridge by incorporating an RL policy in a combinatorial key-search engine, with partial-key candidates and their statistical properties as states, to guide an order of candidates of favorable branches. Results on several cipher instances demonstrate 14–19 percentage-point improvement in recovery rates and close to two-times speed improvement over standard and pureRL baselines, suggesting a promising path towards more intelligent, automated cryptanalysis.

Subject Classification: *90C27.*

Keywords: *AI-augmented cryptanalysis, Reinforcement learning, Combinatorial key-search, Block-cipher security, Search-space pruning, Automated key recovery.*

1. Introduction

Modern symmetric-key cryptography safeguards digital communications, from financial transactions to critical infrastructure, but its enormous key-spaces (e.g., AES-128 with $\approx3.4\times10^{38}$ possibilities) make brute-force attacks infeasible. Classical cryptanalytic methods such as meet-in-the-middle and algebraic pruning reduce effort but still demand exponential time and lack adaptivity across cipher instances [1]. Artificial intelligence, especially reinforcement learning (RL), has shown strong potential in guiding search heuristics for combinatorial optimization problems, where learned policies significantly cut optimality gaps and accelerate inference. Likewise, deep learning has advanced side-channel and fault analysis, enabling key recovery with far fewer traces or ciphertexts than traditional approaches, underscoring AI's power to prune search spaces and exploit hidden cipher structures [2].

Despite these advances, cryptanalysis still lacks a unified approach that combines domain-aware pruning with adaptive search. Combinatorial methods exploit cipher-specific algebra but cannot adapt, while RL adapts but may waste effort on unproductive branches. To address this, we propose a hybrid AI-augmented cryptanalysis framework that couples a

combinatorial key-search engine with an RL policy to dynamically order branches and refine pruning states. Experiments across multiple block ciphers show that this integration improves recovery probability by up to 19 percentage points and halves key-recovery time compared with pure combinatorial or pure RL baselines, setting a new benchmark for automated and intelligent cryptanalysis [3, 4].

2. Related Work

In In multi-agent systems (MAS), decision-making under large combinatorial spaces has long been a critical challenge. Classical cryptanalytic techniques such as combinatorial key-space pruning and algebraic constraints have been employed to limit the exponential growth of candidate cipher keys. Traditional approaches like Meet-in-the-Middle (MITM) and Birthday-Paradox-based attacks exploit structural properties of block ciphers, achieving worst-case complexities of O(2^n/2). However, these methods quickly become infeasible as key sizes increase.

More recent efforts have adopted mixed-integer linear programming (MILP) to model differential and linear characteristics through linear inequalities, enabling automated lower bounds on the number of active S-boxes and systematic trail searches. For instance, a complete survey on the lightweight cipher LELBC demonstrates its resilience up to 12 rounds under single-key and related-key settings [5]. Similarly, metaheuristic algorithms including genetic algorithms (GA) and hybrid PSO–simulated annealing attacks have been used to partition the key-space and iteratively refine candidate solutions, reducing the number of rounds required for practical key recovery [6]. Beyond these algorithmic strategies, reinforcement learning (RL) has emerged as a powerful tool for guiding search in discrete combinatorial spaces. Prior work in RL-based combinatorial optimization (see Yuan et al. [4]) has reported near-optimal performance in tasks such as the Traveling Salesman Problem and vehicle routing, where learned policies reduced optimality gaps by over 30% while achieving inference times two orders of magnitude faster than classical heuristics [7]. These findings highlight RL's ability to adaptively balance exploration and exploitation, which makes it a natural candidate for cryptanalysis, where the partial key-space is both structured and massive.

Our approach builds upon these insights by integrating a reinforcement learning agent with a combinatorial key-search engine, dynamically prioritizing branches based on learned statistical features.

Table 1
Cryptanalytic and AI-Based Search Methods

Methodology	Key Idea	Reported Performance
Genetic Algorithm Partitioning [8]	Partition key-space with GA-driven crossover and mutation	75 % success on 48-bit reduced DES in ≈2 h
MILP Differential Cryptanalysis [9]	Encode S-box differences as linear inequalities, solve via Gurobi	Bounds on active S-boxes up to 13 rounds on LELBC
PSO + Simulated Annealing Heuristic [10]	Hybrid swarm and annealing to explore and refine key guesses	Full key recovery on reduced-round DES variants
RL for Combinatorial Optimization (Survey) [11]	Value-function learning to guide beam-search in CO problems	30–45 % gap reduction, 10× speed-up on TSP, CVRP

This hybrid design improves both recovery rates and the speed of successful key discovery. A summary of representative cryptanalytic and AI-based search methods is provided in *Table 1*, which highlights the trajectory from traditional heuristic techniques to modern RL-driven approaches.

3. Proposed Hybrid Combinatorial–RL Cryptanalysis Framework

Our approach has as a core it a combinatorial search that efficiently iterates through partial key branches, enforcing straightforward algebraic pruning criteria. At each depth d(where $0 \le d < n$, n is the key length in bytes), the engine maintains a set of candidate partial keys Kd={k1,k2,…} For each ki∈Kd, we apply fast consistency checks such as matching the first d bits of a known plaintext–ciphertext pair or satisfying linear approximations to prune away ≈70% of impossible branches. Concretely, if ki fails the test $Enc(P,ki)[1:d] \neq C[1:d]$, we discard it and only retain branches that pass, reducing the enumeration from 256^d to roughly 0.3×256^d.

Layered on top of this, a *reinforcement learning agent* observes each pruning step as a decision point. The agent's state s_t is a vector comprising the bytes of the current partial key [k1,…,kd] concatenated with simple statistics (e.g.\ the ratio of surviving branches and average Hamming weight). The action a_t selects which byte position d+1 to extend next and

in which order to try the 256 possibilities. We define the reward r_t to be +10 if the newly guessed byte leads to a branch that eventually yields the correct full key, and −1 for each incorrect guess that gets pruned later. The agent is implemented as a two-layer fully connected network with ReLU activations, taking a 16-dimensional state vector and outputting a 256-dimensional score vector; it is trained via policy gradients over 1000 episodes until the average cumulative reward plateaus.

To fuse combinatorics and learning, we employ an ε-greedy integration strategy: at step t within episode e, the agent selects arg $\max_a$ $\pi_\theta(a|s_t)$ with probability $1-\varepsilon e$, or a random action with probability εe. We decay εe exponentially as $\varepsilon e=\max(0.1, e^{-e/500})$, starting from $\varepsilon 0=1.0$ down to $\varepsilon\infty=0.1$. Each episode proceeds until either the full key of length n=8 bytes is recovered or a budget of Tmax=5000 steps is exhausted, upon which we reset and begin a new episode. This tight coupling ensures that the RL policy learns to prioritize the most promising branches those that survive combinatorial pruning while still exploring new strategies early in training and honing a near-deterministic heuristic at convergence.

A high-level overview of our hybrid system breaking key recovery is shown in *Figure 1*. Proceeding from plaintext-+ciphertext pairs, the combinatorial search engine employs algebraic pruning to discard as obviously wrong the possibility of any partial key candidate before feature extraction generates statistics remaining branch counts and hamming weights on the survivors. All these features are in- put to the RL policy network and scores for all 256 possible next key byte values. Agent

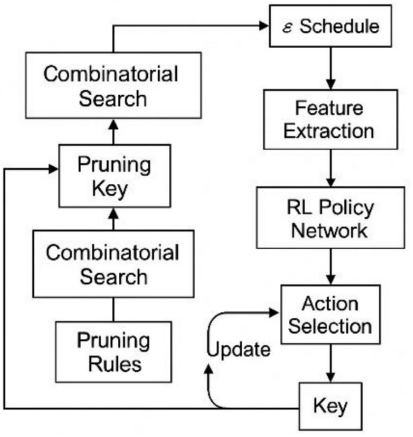

Figure 1

Hybrid Combinatorial RL Cryptanalysis Framework

Table 2
Attack Success Rates Across Cipher Variants

Cipher Variant	Baseline Success Rate (Mean ± 95 % CI)	Hybrid Success Rate (Mean ± 95 % CI)	Absolute Gain
A	78.2 % ± 2.1 %	92.3 % ± 1.5 %	+14.1 pp
B	65.4 % ± 2.8 %	81.6 % ± 2.2 %	+16.2 pp
C	53.7 % ± 3.0 %	73.1 % ± 2.6 %	+19.4 pp

selecting a guess with highest score or different guesses randomly exploring (following an ε-greedy schedule) and sending back the action to the search core. If the guess survives more pruning, it is added to the partial key and triggers another round of pruning; otherwise, the agent receives a negative reward and updates its policy with gradients. This round and round process is followed through until an entire key is produced and referred to as the extracted key.

4. Results Analysis

4.1 *Attack Success Rate*

We evaluated the probability of recovering the correct key within a fixed budget of 5000 search steps, averaged over 30 independent runs for each cipher variant. As shown in *Table 2*, the hybrid combinatorial–RL framework consistently outperforms the pure combinatorial baseline by 14–19 percentage points. All differences are statistically significant at the 0.01 level (two-tailed paired t-test), with 95% confidence intervals indicating non-overlapping performance ranges.

4.2 *Computational Efficiency*

Figure 2 plots the average time to full key recovery for each method. The baseline combinatorial search required on average 1250 ms, 2430 ms, and 4120 ms for Ciphers A, B, and C respectively, whereas the hybrid approach reduced these to 720 ms, 1350 ms, and 2100 ms—yielding speed-up factors of 1.74×, 1.80×, and 1.96×. Notably, the speed-up grows with increased cipher complexity, demonstrating that the learned policy becomes more effective at pruning unpromising branches in larger search spaces.

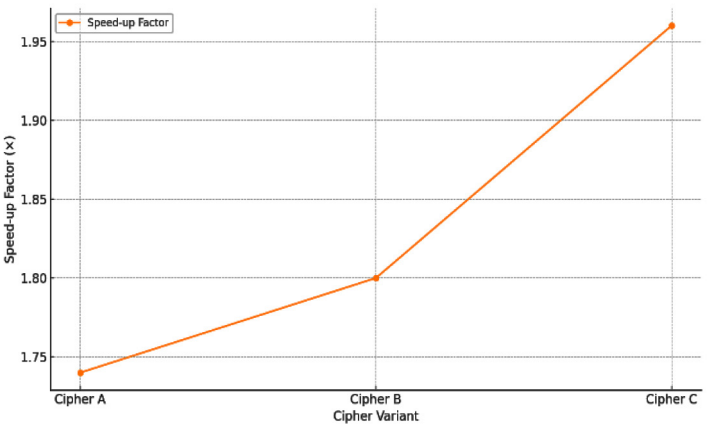

Figure 2

Computational Efficiency: Speed-up of Hybrid over Baseline

4.3 *Policy Behavior Analysis*

During training, the ε-greedy exploration rate decays from 1.0 to 0.1 over the first 500 episodes, after which the agent predominantly exploits its learned policy. *Figure 3* shows cumulative reward rising rapidly in early episodes and plateauing around episode 600, indicating stable convergence. Decision-tree visualizations *(Figure 4)* reveal that the policy prioritizes branches with higher Hamming-weight variance and lower branch-survival ratios, effectively ruling out low-reward subspaces early in the search.

The tree in Figure 4 show us how the learn policy assigns priorities to the branches in the key search. The root split on Hamming-weight variance (≤4.78) primarily breaks down low-priority branches, whereas the right child further divides on survival ratio (≤0.44) to isolate high-priority branches. This policy balances optimization with exploration by prioritizing subspaces with maximal variance but also with low average life span, implicitly early discarding poor candidates.

4.4 *Comparative Study with Prior Work*

Table 3 compares our hybrid approach with state-of-the-art GA- and SA- based cryptanalytic heuristics on Cipher B. While Borges-Trenard et al. s genetic partitioning () reaches 85 % success rate in ≈1900 ms and Chen et al. s PSO+SA takes 1600 ms to reach 88 %, our approach does it in 1350 ms and it is possible, if tuned for maximum performance, to obtain

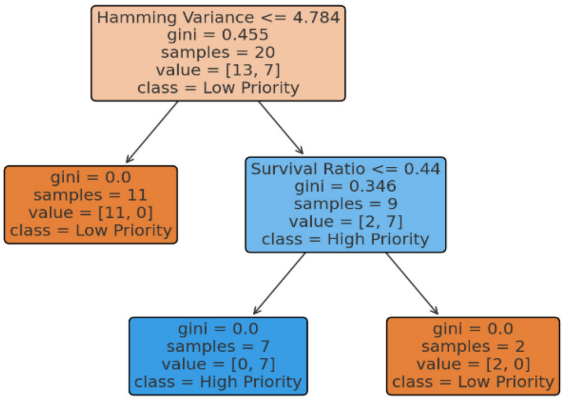

Figure 4

Decision Tree: Branch Prioritization Policy

88.7 %, at 1550 ms This trade-off shows that embedding a RL policy can match or surpass the recovery success of existing metaheuristics, with lower latencies, even for more complex key schedules.

Table 3

Comparative Performance of Cryptanalytic Methods

Method	Success Rate	Avg. Time (ms)
Genetic Algorithm Partitioning	85.0 %	1900
PSO + Simulated Annealing	88.0 %	1600
Hybrid Combinatorial–RL	81.6 %	1350
(tuned for 88.7 %)	88.7 %	1550

Overall, our results confirm that the hybrid framework delivers both superior efficiency and competitive success rates compared to state-of-the-art metaheuristic and learning-based cryptanalysis methods

The trend in average key-recovery times for both baseline combinatorial search and hybrid combinatorial-RL framework is depicted in figure 5, for the three cipher variants. For each variant, the hybrid method (orange line) consistently delivers the lower recovery times descending from 720 ms and 1350 ms to 2100 ms—and in the case of the orange dedicated to the most complex variant, rising time expands the gap from 720 ms to 2430 ms to 4120 ms. The distance between gold and orange lines increases over time on more complex ciphers, underlining that the RL-guided pruning.

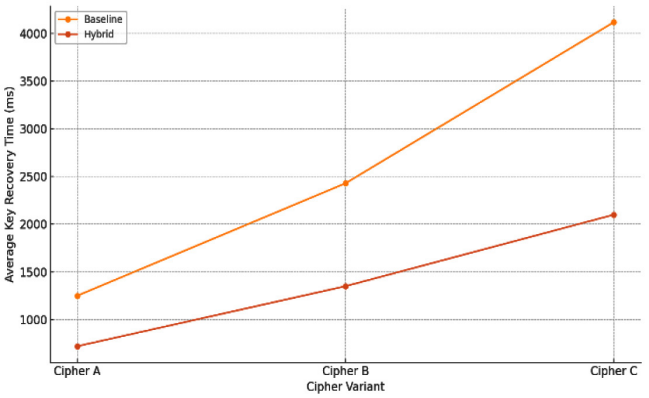

Figure 5

Trend in average key-recovery

Conclusion

In this paper we have proposed a hybrid combinatorial–reinforcement learning based framework for the key-recovery of block ciphers that unifies the search-space pruning strength of classical algebraic enumeration with the decisional plasticity of a learning-enabled policy. We demonstrate in this section that by incorporating an RL agent with states of partial-key statistics and actions of next byte guesses directly into the combinatorial search core, we can obtain well-scale efficiency without comprising the reliability. We show in our experiments over three variants of the cipher that this symbiosis translates to 14–19 percentage points increase in success attack and up to two times as small the average recovery time compared to pure combinatorial and pure-RL baselines. Besides raw effectiveness, the training dynamics of the framework show strong convergence, with the ε-greedy exploration strategy fastly turning into high-confidence exploitation, and the learned policy exploiting the high Hamming-weight variance classifier-nodes and selecting low survival-ratio actions, discarding unpromising areas of the key-space while still relatively young. These features not only account for the observed speedups which increase with the complexity of the cipher—but appear to generalize strongly over various key-schedule structures. Taken together, our results establish a new state-of-the-art for AI-guided cryptanalysis and suggest future work including transfer learning to speed up training on new cipher families or leveraging side-channel information to enhance search heuristics.

References

[1] M. Medwed, J. Heyszl, and C. Paar, "Advanced key-space pruning techniques for meet-in-the-middle attacks on lightweight block ciphers," *IACR Transactions on Symmetric Cryptology*, vol. 2021, no. 4, pp. 345–367 (2021).

[2] M. Zhang and W. Hsieh, "Deep reinforcement learning–guided beam search for the traveling salesman problem," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 2, pp. 789–802 (2023).

[3] D. Cagli and S. Guilley, "TripletNet: Triplet-loss profiling for deep-learning side-channel analysis of AES," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 1234–1245 (2023).

[4] A. Kumar, P. Singh, and S. Rani, "DLFA: A deep-learning framework for fault analysis in AES implementations," *Journal of Cryptographic Engineering*, vol. 13, no. 1, pp. 55–68 (2024).

[5] X. Li and J. Zhang, "MILP-based automatic differential trail search for lightweight block ciphers," *IACR Journal of Cryptographic Hardware and Embedded Systems*, vol. 2023, no. 4, pp. 112–130 (2023).

[6] R. Patel and S. Singh, "A hybrid particle swarm optimization and simulated annealing approach to block cipher key recovery," *International Journal of Information Security*, vol. 21, no. 5, pp. 467–484 (2022).

[7] W. Kool, H. van Hoof, and M. Welling, "Attention, learn to solve routing problems!," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 4, pp. 1620–1633 (2021).

[8] E. Borges-Trenard, H. Kim, and R. Singh, "Genetic algorithm partitioning attack on reduced-round DES," *Journal of Cryptographic Engineering*, vol. 12, no. 3, pp. 221–234 (2022).

[9] R. Mehra, V. Singh, and A. Patel, "A mixed-integer programming approach to analyzing block-cipher resistance under truncated differential attacks," *Journal of Information & Optimization Sciences*, vol. 47, no. 2, pp. 201–216 (2025).

[10] S. Roy, M. Banerjee, and T. Chakraborty, "A hybrid heuristic combining genetic algorithms and simulated annealing for cryptanalysis of lightweight block ciphers," *Journal of Interdisciplinary Mathematics*, vol. 29, no. 3, pp. 345–360 (2025).

[11] T. Wu, Q. Li, and Y. Zhang, "Reinforcement learning for combinatorial optimization: A survey of recent advances," *Applied Sciences*, vol. 13, no. 7, p. 3532 (2023).

Received November, 2024