MACHINE-LEARNING-BASED SELECTION OF ENCRYPTION ALGORITHMS FOR UAV

Yuliia Polischuk, Dmytro Proskurin, Tetiana Hryniuk

Abstract. *Unmanned Aerial Vehicles (UAVs) generate heterogeneous data streams, including real-time video, telemetry, and command-and-control messages, while operating under strict constraints on communication bandwidth, energy consumption, computational resources, and mission risk levels. The use of a single universal encryption algorithm across all UAV scenarios is inefficient, as it either results in excessive computational and energy overhead or provides an insufficient level of cryptographic protection in critical conditions.*

In this work, a set of eight criteria (K_1–K_8) for the informed selection of symmetric encryption algorithms is formalized. Based on these criteria, a machine learning (ML)-driven approach is proposed to enable the automated selection of the most appropriate cryptographic algorithm according to the parameters of a given UAV deployment scenario.

The key scientific contribution lies in the development of a UAV-oriented dataset that integrates mission context parameters (operational conditions, resource constraints, threat levels) with cryptographic characteristics of algorithms (security strength, performance, and resource efficiency). The proposed feature structure provides a formalized representation of the algorithm selection problem and establishes a foundation for training intelligent decision-making models.

Each feature group is grounded in state-of-the-art research in UAV cybersecurity, lightweight cryptography, and ML-based adaptive encryption. Furthermore, this work extends the authors' previous studies related to the development of cryptographic algorithm datasets and the application of neural networks for ensuring image confidentiality in UAV systems.

Keywords: *Unmanned Aerial Vehicles (UAVs), post-quantum cryptography, machine learning, adaptive encryption, cryptographic algorithm selectio.*

Introduction

Modern UAVs are increasingly used in reconnaissance, search-and-rescue, critical infrastructure monitoring and combat support. They transmit high-resolution video, sensor data and control commands over wireless channels that are often unstable, bandwidth-limited and exposed to active adversaries. Recent studies on UAV and drone-network security emphasize that robust confidentiality and integrity are mandatory for these systems but also stress the severe resource constraints of aerial platforms and the inadequacy of "desktop-grade" cryptography when used without adaptation [2], [4-7].

At the same time, symmetric encryption offers a wide range of block and stream ciphers, modes of operation and parameter choices. Studies on lightweight cryptography for IoT and UAVs show that algorithm performance, memory footprint and energy consumption can differ by orders of magnitude on microcontrollers, while security margins and standardization levels also vary significantly [6], [16].

Manually selecting an algorithm for every UAV mission profile quickly becomes complex and error prone. To address this, prior work by Gnatyuk et al. proposed a universal dataset of cryptographic algorithms and used an artificial neural network to select the optimal algorithm for ensuring the confidentiality of UAV images, based on speed, cryptographic strength and other parameters [1].

In this article we extend that line of work. Using an internally developed framework of eight selection criteria K_1–K_8 (block size, key size, resistance to attacks, transparency, weak keys, performance, memory requirements, and usage/licensing constraints), we: map these criteria to concrete UAV-relevant conditions (data type, link quality, mission criticality, platform class); design a UAV-specific dataset that encodes both scenario features and algorithm properties; justify, with explicit references, that each group of dataset features is meaningful and already used (implicitly or explicitly) in the literature on UAV security and lightweight cryptography; outline how this dataset can be used to train ML models for automatic algorithm selection.

Related Work

UAV security and lightweight cryptography

Recent studies on UAV communication security and UAV/FANET security consistently identify confidentiality, integrity and availability of wireless links as key challenges and highlight that small UAVs are extremely resource-constrained [2, 4-7].

Parallel studies on lightweight cryptography for IoT and edge devices provide detailed taxonomies of block ciphers, stream ciphers and authenticated encryption schemes optimized for low memory, low energy and low latency operation. These works show that throughput, RAM/ROM consumption, power consumption and hardware acceleration support are standard evaluation metrics when comparing algorithms [5-6, 16-17].

Specific to UAVs, several papers evaluate lightweight schemes such as ASCON and other NIST LWC candidates in simulated UAV networks, demonstrating how algorithm choice affects energy use and communication delay [5-6].

These findings align directly with our criteria K_6(performance) and K_7(memory/resource requirements) and motivate their inclusion as dataset features.

Machine learning for adaptive encryption

There is a growing body of work that uses ML to adapt cryptographic mechanisms to context:

- Gnatyuk et al. construct a dataset of cryptographic algorithms for UAV image confidentiality and train a neural network to select the optimal algorithm based on algorithm speed, security and other parameters [1].
- Adaptive hybrid cryptographic frameworks for IoT dynamically select algorithms and parameters using ML to balance security level and resource usage in real time [14, 17].
- AI-based encryption frameworks for telecom and SDN dynamically select algorithms and keys using machine learning or genetic algorithms to meet confidentiality requirements under performance constraints [14].
- Reinforcement learning approaches adjust encryption strength or algorithm choice based on observed network conditions and energy budget [15].

These works confirm two key assumptions behind our dataset design:

- Algorithm choice can be modelled as an ML classification problem driven by measurable features (speed, security, resource use, context).
- UAV-specific conditions (link quality, device constraints, data type) are legitimate features for such models, not just purely cryptographic parameters.

Criteria K_1–K_8 for Algorithm Selection

The internal framework defines eight criteria that quantify the suitability of an encryption algorithm for a given scenario:

- K_1: Block size. Values such as 64, 96, 128, 160, 192, 256 bits. Larger blocks help mitigate birthday-bound issues and Sweet32-type attacks and are preferred for large volumes and long-term storage, as noted in Sweet32 and in BSI guidance against 64-bit block ciphers [8, 13]. In UAV contexts that involve continuous video, sensor streams, or long-term encrypted storage, block sizes of 128 bits or more are preferred. This criterion measures not only the block size itself but whether the cipher avoids structural limits that appear when block sizes are too small for the expected data volume.
- K_2: Key size. Key lengths 128/192/256 bits and the number of supported key sizes, motivated by NIST SP 800-57 security strength levels and the AES standard [9-10]. Algorithms that only support short or fixed keys are less adaptable, particularly in environments with evolving threat models or where keys may need to stay valid for extended mission lifetimes.
- K_3: Resistance to cryptanalytic attacks. This criterion evaluates the robustness of an algorithm against major classes of attacks: differential and linear cryptanalysis, related-key attacks, biclique reductions, algebraic attacks, statistical distinguishers, and side-channel attacks on practical implementations. Each algorithm is graded on a three-point scale: unstable (known practical or near-practical breaks), partially stable (reduced-round weaknesses or borderline margins), or stable (no known feasible attacks on full-round configurations). In UAV deployments where adversaries may intercept large data volumes or deploy advanced hardware, only ciphers with no practical breaks across all rounds and modes are considered reliable. The evaluation includes both theoretical analysis and empirical evidence from the literature.
- K_4: Transparency of structure. Transparency captures how open the design and implementation are. Fully open algorithms include complete specifications, reference implementations, test vectors, and public scrutiny. Partially open designs may reveal the algorithm but restrict code, licensing, or test suites. Closed or proprietary designs limit inspection and hinder formal analysis. For UAV security, where certification, reproducibility, and interoperability matter, transparency supports trust and reduces the risk of embedded weaknesses or undisclosed assumptions. Algorithms with an open design and broad peer review score highest, while proprietary or nationally restricted ciphers score lower.
- K_5: Weak/equivalent keys. Some historical ciphers contain weak keys that produce predictable patterns, or equivalent keys that generate identical encryption behavior despite being different values. This criterion assesses whether such key classes are known, how many exist, and whether they impact real-world deployments. Modern designs aim to eliminate weak-key structures entirely, ensuring uniform behavior across the whole key space. UAV systems that cannot afford intermittent rekeying or large key-management overheads are especially sensitive to this issue, since a weak or equivalent key can compromise privacy for the duration of a mission. Algorithms with no known weak or equivalent keys receive the highest score.
- K_6: Performance (throughput). Performance is measured in effective throughput on representative UAV hardware. Since UAVs range

from lightweight microcontrollers to high-end embedded boards, throughput is categorized as low (≤10 Mb/s), medium (10–100 Mb/s), or high (≥100 Mb/s). These values reflect real constraints such as real-time video encryption, telemetry bandwidth, and CPU load budgets. High-throughput designs are essential for continuous video streams or dense sensor fusion, while low-throughput algorithms may still be viable for low-data-rate control links or periodic telemetry. This criterion integrates algorithmic complexity, platform-specific optimizations, and the presence of hardware acceleration.

- K_7: Memory requirements. Memory consumption includes RAM usage during operation and ROM/flash footprint for lookup tables, S-boxes, and code. Requirements are classified as low (<1 KB), medium (1–10 KB), or high (>10 KB). This reflects the diversity of UAV compute platforms: from 8/16-bit MCUs with strict memory limits to 32-bit Cortex-M devices and higher-end boards capable of running full protocol stacks. Algorithms with large S-boxes or multi-stage key schedules may exceed the memory budgets of resource-constrained UAVs. Compact algorithms that maintain security under tight memory conditions score highest.
- K_8: Usage and licensing constraints. This criterion examines legal, regulatory, and licensing conditions that affect deployment. Open-source, royalty-free algorithms receive the highest score, as they allow integration without cost or restrictions. Partially restricted designs may impose usage limits, patents, export requirements, or require specific certifications such as FIPS 140-3. Proprietary or nationally restricted algorithms often limit international deployment, reduce interoperability, and complicate long-term maintenance. For UAV systems that operate across jurisdictions or rely on commercial off-the-shelf components, these constraints can determine whether an algorithm is practical or deployable.

Internal tables map these criteria to UAV-relevant conditions such as "real-time video over unstable link," "high-risk mission with long-term storage," "weak MCU with low data volume," and specify recommended K-values for each case. This structure already resembles a feature space: each mission scenario is represented by K-values and additional context.

A multicriteria analysis based on these eight criteria is shown in Table 1, where each algorithm receives a binary score for K1 through K8 consistent with the definitions above.

Table 1

A multicriteria analysis

Algorithms/Criteria	K1	K2	K3	K4	K5	K6	K7	K8
Symmetric, block encryption								
AES	+	+	+	+	+	+	+	+
DES	-	-	-	+/−	-	-	+	+
3DES	-	-	+/−	+/−	+	-	+	+
CAST-128	-	-	+	+	+	+	+	+
Blowfish	-	+	-	+	-	-	+	+
Symmetric, stream encryption								
RC4	+	+	-	+	-	+	+	+
OTP	+	-	+	+	+	+	+	+
ChaCha20	+	-	+	+	+	+	+	+
ChaCha20-Poly1305	+	-	+	+	+	+	+	+
Asymmetric encryption								
RSA	N/A	+	+	+	+	-	+/−	+
ECC	N/A	+	+	+	+	-	+/−	+
Hash function								
SHA-256	+	-	+	+	+	+	+	+

The multicriteria evaluation in Table 1 shows that no single algorithm satisfies all eight criteria. AES and ChaCha20-based constructions achieve the most balanced profiles, scoring positively across security, transparency, memory use, and deployment flexibility. Legacy block ciphers such as DES and 3DES fall short due to weak security margins and low performance, while Blowfish and CAST-128 offer mixed results tied to block size or key structure. Stream ciphers like RC4 are penalized for known cryptanalytic weaknesses, whereas OTP scores well in principle but is impractical due to key-management demands. Asymmetric schemes (RSA, ECC) provide strong security and transparency but incur heavier performance costs, reflected in lower throughput and higher memory impact. SHA-256 fulfills most criteria except key-size flexibility, which is not applicable to hash functions. Overall, the table highlights tradeoffs and confirms that algorithm selection must be scenario-driven rather than universal.

Candidate Algorithms and Scenario-Based Recommendations

Using the criteria above, we can define a mapping from typical UAV scenarios to recommended algorithms. The candidate set includes:

- AES in different modes (e.g., AES-CTR, AES-GCM);
- conservative AES alternatives such as Serpent and Camellia;
- modern stream/AEAD ciphers such as ChaCha20-Poly1305;

- lightweight ciphers and NIST LWC candidates (e.g., ASCON, SIMON, SPECK) [4-6, 16].

We summarize representative scenarios (simplified):

1. Real-time video, medium threat, powerful CPU with AES acceleration.

- Data type: 1080p video stream.
- Requirements: high throughput, low latency, medium-to-high confidentiality, security lifetime of a few years, FIPS preference.
- K-targets: K1 = 128, K2 = 128, high K6, high K7, strict K8.
- Recommended algorithm: AES-CTR-128 or AES-GCM-128 [9, 11-12].

2. Top-secret mission, poor link, long-term storage of recorded imagery.

- Data type: compressed images and video logs.
- Requirements: very high confidentiality, long security lifetime (decades), possibly limited bandwidth.
- K-targets: larger K1/K2 (128/256 bits), maximal K3, high K4, strict K8.
- Recommended algorithms: Serpent-256 or Camellia-256, offering strong security margins and standardization support [4, 16].

3. Nano-UAV C2 channel on an 8-bit MCU.

- Data type: short C2 messages.
- Requirements: very small RAM and Flash, high latency sensitivity, relatively short secrecy lifetime.
- K-targets: modest K1, K2 = 128, low K7, relaxed K8.
- Recommended algorithms: SIMON-64/128, ASCON-128a or similar lightweight ciphers [4-6].

4. Web-like telemetry traffic, high speed, no AES hardware, open-source preference.

- Data type: telemetry and control messages, possibly over a TCP/UDP-based VPN.
- Requirements: good throughput on general-purpose CPU, strong AEAD, open and patent-free.
- K-targets: K1 = 128 or stream, K2 = 256, high K3, high K4, relaxed K8 (FOSS).
- Recommended algorithm: ChaCha20-Poly1305, widely deployed in IETF protocols when AES acceleration is absent [3, 16, 19].

These rules provide an "expert label" for each scenario - exactly the kind of mapping that an ML model can learn to reproduce and generalize.

Dataset Design for ML-Based Algorithm Selection in UAV Systems

The ML task we solve is a supervised classification problem: given a description of a UAV communication scenario (the input), the model predicts which encryption algorithm (and parameters) should be used (the output label). This follows the standard pattern in machine learning where we learn a mapping from feature vectors to discrete classes [23]. At the same time, from a meta-learning point of view, it is a specific case of the algorithm selection problem [19-21].

The dataset therefore has two sides:

1. A scenario description (what data, which UAV, which environment, which security needs);
2. An algorithm profile (what this cipher can do, how secure and fast it is on UAV hardware);
3. A label that says whether this algorithm is optimal/acceptable/infeasible for this scenario.

Input representation: UAV scenario feature vector

The input to the ML model is an encoded description of the current UAV situation. We structure this into four groups of features, consistent with how UAV security and lightweight cryptography studies describe systems [2, 4–7, 16]:

1. Mission and data characteristics
2. Communication channel and environment
3. Platform and resource profile
4. Security and regulatory requirements

In simple terms, the model "sees" something like the following for each scenario:

1. Mission & data

a. data_type ∈ {video_stream, still_image, telemetry, c2, log_file}

b. avg_throughput_mbps (e.g., 0.05, 1.2, 15.0)

c. max_burst_mbps

d. latency_sensitivity ∈ {low, medium, high}

e. confidentiality_level ∈ {low, medium, high, top_secret}

f. retention_time_years (how long the data must remain secret)

2. Channel & environment

a. link_stability ∈ {good, medium, poor} or a numeric loss percentage

b. bandwidth_class ∈ {low, medium, high}

c. topology ∈ {single_hop, relay, mesh}

d. jamming_risk ∈ {low, medium, high}

3. Platform & resources

a. cpu_class ∈ {mcu_8bit, cortex_m, cortex_a, x86}

b. has_aes_accel ∈ {0, 1}

c. ram_kb

d. flash_kb

e. energy_class ∈ {ultra_constrained, constrained, relaxed}

4. Security & regulation

a. attack_model ∈ {basic, advanced}

b. security_lifetime_years

c. require_fips ∈ {0, 1}

d. require_national_crypto ∈ {0, 1}

e. open_source_only ∈ {0, 1}

Additionally, we include K-targets that represent the desired cryptographic properties derived from these fields and from the K1–K8 tables:

- K1_block_size_target (e.g., 128)
- K2_key_size_target (e.g., 256)
- K3_resistance_target ∈ {unstable, partial, stable}
- K4_transparency_target ∈ {low, medium, high}
- K5_weak_keys_allowed ∈ {0, 1}
- K6_perf_target ∈ {low, medium, high}
- K7_memory_target ∈ {low, medium, high}
- K8_licensing_target ∈ {strict, moderate, relaxed}

This reflects how standards such as NIST SP 800-57 link security strength, key sizes and lifetimes to the application context, and how UAV-focused work derives performance and resource constraints from mission types [2, 4-7, 10, 13, 16].

Example 1 – High-resolution video from a tactical UAV

A fixed-wing UAV sends 1080p video to a ground station over a moderately unstable link. It has a Cortex-A CPU with AES acceleration. The mission is sensitive but not top-secret; video should remain confidential for about 3 years. The feature vector could include:

- data_type = video_stream
- avg_throughput_mbps = 15.0
- latency_sensitivity = high
- confidentiality_level = high
- retention_time_years = 3
- link_stability = medium
- bandwidth_class = medium
- cpu_class = cortex_a
- has_aes_accel = 1
- energy_class = constrained
- security_lifetime_years = 3
- require_fips = 1

Derived K-targets:

- K1_block_size_target = 128
- K2_key_size_target = 128
- K6_perf_target = high
- K7_memory_target = high
- K8_licensing_target = strict

Example 2 – Nano-UAV C2 channel

A palm-sized quadrotor has an 8-bit MCU and sends only short control messages. The link has low bandwidth but is reasonably stable. Commands are very sensitive but only need confidentiality for minutes or hours.

- data_type = c2
- avg_throughput_mbps = 0.02
- latency_sensitivity = high
- confidentiality_level = high
- retention_time_years = 0.1
- link_stability = good
- bandwidth_class = low
- cpu_class = mcu_8bit
- has_aes_accel = 0
- ram_kb = 2
- flash_kb = 16
- energy_class = ultra_constrained
- require_fips = 0
- open_source_only = 1

Derived K-targets:

- K1_block_size_target = 64 or 96
- K2_key_size_target = 128
- K6_perf_target = low
- K7_memory_target = low
- K8_licensing_target = relaxed

These examples show that the input vector is a structured, numeric representation of real UAV conditions, consistent with how UAV security and lightweight crypto papers describe systems [2, 4-7, 16].

Representation of algorithms and K-criteria

The second half of the dataset is the algorithm profile. Lightweight cryptography studies and UAV-specific studies commonly compare algorithms along dimensions of block/key size, security margin, speed, memory footprint and implementability on constrained hardware [4-6, 16]. For each algorithm–mode pair (e.g., AES-CTR-128, ChaCha20-Poly1305, ASCON-128a, SIMON-64/128) we define features such as:

- alg_family ∈ {AES, Serpent, Camellia, ChaCha20, ASCON, SIMON, SPECK, …}
- mode ∈ {CTR, GCM, CBC, stream, AEAD_native}
- block_size_bits (for block ciphers)
- key_size_bits (or list of supported key sizes)
- K3_resistance_score ∈ {0, 1, 2} for unstable / partial / stable
- K4_transparency_score ∈ {0, 1, 2} for low / medium / high (public spec, standardization)
- has_known_weak_keys ∈ {0, 1} (K5)
- throughput_mbps_cortex_a, throughput_mbps_cortex_m, throughput_mbps_mcu8 (K6)
- ram_usage_bytes, flash_usage_bytes (K7)

- is_fips_approved, is_national_standard, is_open_source, is_patent_free (K8)

These values come from standard documents (AES, NIST SP 800-38A/38D, NIST LWC candidates), security analyses of individual ciphers, and benchmark studies for lightweight algorithms on microcontrollers and UAV systems [4-6, 9-12, 16].

Example 1 – AES-CTR-128 on Cortex-A

- alg_family = AES
- mode = CTR
- block_size_bits = 128
- key_size_bits = 128
- K3_resistance_score = 2
- K4_transparency_score = 2
- has_known_weak_keys = 0
- throughput_mbps_cortex_a = high
- ram_usage_bytes = medium
- flash_usage_bytes = medium
- is_fips_approved = 1
- is_open_source = 1
- is_patent_free = 1

Example 2 – SIMON-64/128 on 8-bit MCU

- alg_family = SIMON
- mode = CTR (or CBC)
- block_size_bits = 64
- key_size_bits = 128
- K3_resistance_score = 1–2
- K4_transparency_score = 1
- throughput_mbps_mcu8 = moderate
- ram_usage_bytes = very low
- flash_usage_bytes = very low
- is_fips_approved = 0
- is_open_source = 1

Such profiles are exactly the type of information already collected in [1, 4-6, 15].

Labels: what we teach the model to predict

The target of the ML model is the algorithm recommendation. From an ML perspective this is a multiclass (or multi-label) classification problem [22].

We consider two label designs:

1. Single best algorithm label

a. label_best_algorithm ∈ {AES-CTR-128, AES-GCM-128, Serpent-256, Camellia-256, ChaCha20-Poly1305, ASCON-128a, SIMON-64/128, …}

2. Structured label (preferred)

a. label_alg_family ∈ {AES, ChaCha20, ASCON, SIMON, …}

b. label_mode ∈ {CTR, GCM, stream, AEAD_native}

c. label_key_size_bits ∈ {128, 192, 256, scheme-specific}

Additionally, for each (scenario, algorithm) pair we can define:

- label_feasibility ∈ {infeasible, acceptable, optimal}
- where:
- infeasible: violates at least one hard constraint (e.g., block size below K1 target, not FIPS-approved where required, known weak keys);
- acceptable: satisfies constraints but is not the first choice;
- optimal: matches the expert recommendation (such as those in Section 4).

This mirrors previous work where each algorithm is scored on performance and security and a neural network is trained to reproduce the human expert's choice for UAV image encryption [1].

Constructing the supervised dataset

With the scenario features, algorithm profiles and labels defined, we can build the dataset:

1. Enumerate scenarios

a. Start from hand-crafted UAV scenarios (Section 4) and expand using typical use cases from UAV security studies: surveillance, delivery, swarms, beyond visual line-of-sight (BVLOS) monitoring, etc. [2, 4-7].

2. Encode each scenario as a feature vector

a. For every scenario, fill in all mission, channel, platform, security and K-target fields.

3. Combine with algorithms

a. For each scenario, consider each candidate algorithm (AES-CTR-128, AES-GCM-128, ChaCha20-Poly1305, Serpent-256, Camellia-256, ASCON-128a, SIMON-64/128, etc.) and join the scenario features with the algorithm profile to form a single combined feature vector [1, 4-6, 16].

4. Assign labels via expert rules

a. Use the K1–K8-based rule set and scenario-to-algorithm mappings to decide whether each pair is optimal, acceptable, or infeasible. For example, in the high-resolution video scenario, AES-CTR-128 may be "optimal", ChaCha20-Poly1305 "acceptable", and a 64-bit block cipher "infeasible".

5. Balance and augment

a. Ensure that the dataset covers a wide range of UAV types and conditions (from nano-UAVs to large reconnaissance platforms), and optionally augment by slightly varying numeric parameters (bandwidth, CPU frequency, packet loss) to simulate realistic variability [2, 4-7, 16].

The result is a supervised dataset where each row answers the question: "Given this specific UAV mission and communication context, is this

algorithm a good choice, and if so, is it the best choice?"

Standard supervised learning theory then applies: the ML model is trained to approximate the mapping from scenario features to "best algorithm" labels in a way that generalizes to unseen missions [22].

Validity and alignment with existing research

This dataset design is consistent with:

- Dataset-driven algorithm choice for cryptography: prior work on UAV image confidentiality that builds a cryptographic algorithm dataset and trains a neural network to select algorithms based on speed and strength [1].
- UAV-specific and lightweight cryptography studies: where algorithm comparisons depend on device class, bandwidth, mission type and energy budget [2, 4-7, 16].
- Security and key-management standards: which link security requirements and data sensitivity to key sizes and approved algorithms [9-10, 13].
- Algorithm selection and meta-learning theory: which treats algorithm choice as a learning problem over meta-features of tasks [19-21].

Machine Learning Model and Evaluation

Once the dataset is available, the next step is to choose and train an ML model that can predict the best algorithm for new UAV scenarios. We consider a supervised classification function $f(\mathbf{x}) = y$,

where $\mathbf{x}$ is a feature vector (scenario + K-targets and, optionally, algorithm profile) and y is a label ("best algorithm" or "feasibility class").

Because the features mix categorical (e.g., cpu_class, data_type) and numerical (throughputs, RAM, key sizes) variables and the decision boundaries can be nonlinear, several model families are natural choices [22].

Model choices

1. Tree-based ensembles (Random Forest, Gradient Boosted Trees)

- Well suited for heterogeneous features; can model complex interactions.
- Robust with relatively little feature scaling or preprocessing.
- Provide some interpretability (feature importance), which is useful for explaining why a particular algorithm is selected in a given scenario.

2. Feed-forward neural networks (multilayer perceptrons)

- Capture subtle nonlinear relationships and interactions between many features.
- Already used successfully in related work on selecting cryptographic algorithms for UAV image encryption [1].

3. Meta-learning / algorithm selection models

- More advanced approaches from meta-learning treat the problem as learning to choose among "actions" (algorithms) based on meta-features of the task [19-21].
- These are particularly relevant if we later extend the system to support many more algorithms or to tune algorithm parameters automatically.

In practice, one may start with tree-based models or a small neural network; both handle tabular data well and are easy to deploy.

Training procedure

The training pipeline follows standard supervised learning practice [22]:

1. Preprocessing and encoding

- Convert categorical features (e.g., cpu_class, data_type) into numeric form (one-hot encoding or learned embeddings).
- Normalize numeric features (e.g., throughputs, RAM) as needed, especially for neural networks.

2. Train–validation–test split

- Split scenarios into training, validation and test sets.
- Ensure that entire missions or UAV types are held out in the test set to evaluate generalization to new conditions, not just minor variations of seen ones.

3. Training objective

- For single-label "best algorithm" prediction, use cross-entropy loss on algorithm classes.
- For feasibility labels, train a separate model or a multi-task model that simultaneously predicts feasibility and selects among feasible algorithms.

4. Hyperparameter tuning

- Optimize hyperparameters (tree depth, number of trees, network width/depth, learning rate, etc.) using cross-validation or a validation set.
- This is closely related to algorithm configuration problems studied in meta-learning [20-21].

Evaluation metrics

To demonstrate that the ML component is trustworthy and useful, we evaluate it on several dimensions:

1. Algorithm selection accuracy

- Percentage of scenarios where the model exactly matches the expert-defined best algorithm.
- Top-k accuracy (e.g., whether the expert algorithm is among the top 2–3 recommendations) if the system proposes a ranked list.

2. Constraint satisfaction

- Fraction of predictions that violate no hard constraints (block size, key size, security target, licensing, regulatory).
- In a robust design, these constraints are enforced outside the model, but this metric verifies that the model also learns to respect them.

3. Robustness and generalization

- Performance on unseen UAV types and mission profiles (e.g., new microcontroller classes or combinations of data types and link conditions).
- Important for realistic deployment, where exact training scenarios rarely repeat.

4. Explainability

- For tree-based models, feature importance and decision paths can show, for example, that "high throughput + AES hardware + FIPS requirement" strongly favors AES-CTR/AES-GCM, while "8-bit MCU + low RAM" shifts preference to lightweight ciphers.
- Such explanations are crucial in safety-critical and regulated domains.

Extensions: online and adaptive learning

The proposed dataset also enables more advanced approaches:

1. Reinforcement learning / adaptive selection

- RL-based models can be trained to adjust algorithms in real time based on observed latency, energy consumption and packet loss, as explored in adaptive encryption for wireless sensor networks and energy-aware cryptographic frameworks [15-16].

2. Meta-learning for new environments

- As new UAV platforms and algorithms are added, meta-learning methods can quickly adapt the selection model without retraining from scratch, reusing knowledge about how features map to suitable algorithms [20-21].

These directions show that the proposed dataset is not only useful for a single static classifier but can also serve as the foundation for more sophisticated, adaptive cryptographic control.

Integration into a UAV Cryptographic System

To move from theory to practice, the ML-based selector must be embedded into a concrete system architecture. UAV security frameworks typically distinguish between multiple layers (physical, link, network, application) and define where cryptographic functions reside [2, 4-7]. Our system focuses on the link/network/application boundary, where data is prepared for transmission and encryption.

Architectural placement

A simplified architecture includes the following components:

1. Mission planning and configuration: the operator or higher-level system defines the mission: payload type (video, imagery), operational area, expected duration, secrecy level, applicable policies (e.g., FIPS-only, national crypto).

2. Context sensing and collection UAV and ground station monitor:

- link quality (RSSI, packet loss, bandwidth),
- platform state (CPU load, available RAM, battery level),
- current traffic mix (video vs telemetry vs C2).

3. Feature extraction and policy translation converts mission and context information into the feature vector format (Section 5.1), including K1–K8 targets and other fields.

4. Rule-based policy filter 1) encodes non-negotiable constraints:

- FIPS or national standards;
- minimum key sizes from NIST SP 800-57 for given lifetimes [10];
- deprecated or disallowed algorithms [8, 13].

2) Removes from consideration any algorithm that fails these constraints, independently of the ML model.

5. ML-based selector receives the feature vector and the list of remaining algorithms; outputs the recommended algorithm family, mode and key size, and optionally a ranking of alternatives.

6. Cryptographic engine instantiates and configures the chosen algorithm in the communication stack (e.g., AES-GCM in an IPsec-like tunnel, ChaCha20-Poly1305 in a DTLS-like tunnel, ASCON in a lightweight datagram protocol); handles key management in accordance with NIST and system-specific key lifecycle policies [10, 12-13].

In this design, ML augments but does not replace formal policy and cryptographic best practices. Hard constraints and standards are enforced by rules; ML helps choose among remaining policy-compliant options.

When and how often selection is performed

The selection process can operate in several modes:

- Per mission

Algorithm is chosen once during mission planning and remains fixed. Simple, but cannot adapt to changing conditions.

- Per connection or session

Algorithm is chosen when a new secure channel (e.g., video link or C2 link) is established. Allows different algorithms for video vs C2 vs telemetry and can react to significant changes between sessions.

- Per data type / flow

Multiple parallel secure channels with different algorithms can be maintained simultaneously (e.g., AES-GCM for C2, ChaCha20-Poly1305 for telemetry, ASCON for experimental channels).

- Adaptive over time

In more advanced setups, reinforcement learning may periodically re-evaluate and adjust algorithms in response to measured performance and energy metrics, subject to strict protections against downgrade attacks and with careful key management [15-16].

In all cases, protocol design must prevent attackers from forcing insecure downgrades and must preserve forward secrecy where required.

Security and safety considerations

Introducing ML into a security-critical system requires care:

- Defense in depth

ML is only one layer. Cryptographic strength, proper key management, robust implementations and protocol correctness remain essential and are governed by standards such as FIPS 197, NIST SP 800-57 and NIST SP 800-38A/38D [9-13].

- Fail-safe defaults

If the ML model or feature extraction fails, the system should revert to a conservative default (e.g., AES-GCM with strong keys on capable hardware) rather than downgrading encryption.

- Robustness against manipulation

Because the algorithm choice depends on sensed context (bandwidth, RSSI, etc.), an attacker could try to manipulate these signals to force weaker algorithms. Mitigations include: restricting the policy space of allowed algorithms; preferring more secure algorithms when in doubt; logging and auditing algorithm changes.

- Transparency and certification

For systems that must be certified (e.g., in military or critical infrastructure use), it helps that the underlying criteria (K1–K8) and policy rules are explicit and traceable to standards and published research.

By embedding the ML selector in a broader, policy-driven architecture, the proposed system turns the complex multi-criteria choice of encryption algorithm for UAVs into a structured, justifiable and partially automatable process.

Conclusion

This article combined an internally developed set of eight cryptographic selection criteria K1–K8 with recent research on UAV security, lightweight cryptography and ML-based adaptive encryption to design a UAV-specific dataset for training machine-learning models that automatically choose encryption algorithms.

We showed that: 1) each dataset feature group (mission/data, channel, platform, security/regulation, algorithm properties) is justified by existing standards and literature; 2) prior work has already demonstrated the feasibility of such datasets and neural-network-based selection for UAV image confidentiality [1]; 3) the proposed dataset design is a natural extension of existing K-criteria tables and scenario-based algorithm recommendations.

The resulting ML-based selector can be integrated into a UAV cryptographic system as a policy-compliant decision aid that adapts algorithm choice to mission context. Future work can extend this framework to include post-quantum symmetric and hybrid schemes, multi-objective optimization (jointly minimizing latency, energy and detection risk), and full reinforcement-learning approaches that continuously adapt algorithm choices during UAV missions.

Acknowledgement

This work was carried out within the research project "Cryptographic Data Protection System Based on Post-Quantum Encryption Algorithms and Artificial Intelligence." (No. 0125U001588), funded by the Ministry of Education and Science of Ukraine during 2025-2027.

References

[1] S. Gnatyuk, A. Okhrimenko, D. Navrotskyi, D. Proskurin, and B. Horbakha, "Dataset of Cryptographic Algorithms for UAV Image Encryption Based on Artificial Neural Networks," in *Proc. Classic, Quantum, and Post-Quantum Cryptography (CQPC)*, CEUR Workshop Proc., vol. 3504, 2023.

[2] S. Gnatyuk, O. Kliushnyk, D. Navrotskyi, B. Horbakha, and D. Proskurin, "Analysis of Methods for Ensuring the Confidentiality of Data Transmitted from UAVs," *Cybersecurity: Education, Science, Technique*, vol. 1, no. 17, pp. 167–186, 2022.

[3] A. Langley, W. Chang, N. Mavrogiannopoulos, J. Strombergson, and S. Josefsson, "ChaCha20-Poly1305 Cipher Suites for Transport Layer Security (TLS)," RFC 7905, Jun. 2016, doi: 10.17487/RFC7905.

[4] R. Aissaoui, J.-C. Deneuville, C. Guerber, and A. Pirovano, "A Survey on Cryptographic Methods to Secure Communications for UAV Traffic Management," *Vehicular Communications*, vol. 44, art. 100661, Aug. 2023, doi: 10.1016/j.vehcom.2023.100661.

[5] A. Patel, "Analysis of Light-Weight Cryptography Algorithms for UAV-Enabled Networks," *arXiv preprint* arXiv:2504.04063, 2025.

[6] O. Valikhanli and F. Abdullayeva, "Securing UAV Flight Data Using Lightweight Cryptography and Image Steganography," *Int. J. Adv. Comput. Sci. Appl.*, vol. 16, no. 5, pp. 278–288, 2025, doi: 10.14569/IJACSA.2025.0160527.

[7] D. Alsadie, "Cybersecurity and Artificial Intelligence in Unmanned Aerial Vehicles: Emerging Challenges and Advanced Countermeasures," *IET Information Security*, 2025, doi: 10.1049/ise2/2046868.

[8] Karthikeyan Bhargavan and Gaëtan Leurent. 2016. On the Practical (In-)Security of 64-bit Block Ciphers: Collision Attacks on HTTP over TLS and OpenVPN. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16). Association for Computing Machinery, New York, NY, USA, 456–467. https://doi.org/10.1145/2976749.2978423

[9] National Institute of Standards and Technology, *Advanced Encryption Standard (AES)*, FIPS PUB 197, Nov. 2001.

[10] E. Barker, *Recommendation for Key Management—Part 1: General*, NIST Special Publication 800-57, Part 1, Rev. 5, 2020.

[11] M. Dworkin, *Recommendation for Block Cipher Modes of Operation: Methods and Techniques*, NIST Special Publication 800-38A, 2001.

[12] M. Dworkin, *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*, NIST Special Publication 800-38D, 2007.

[13] Federal Office for Information Security (BSI), *Cryptographic Mechanisms: Recommendations and Key Lengths*, BSI Technical Guideline TR-02102-1, Version 2025-1, Mar. 2025.

[14] P. R. Kumar and S. Goel, "A Secure and Efficient Encryption System Based on Adaptive and Machine Learning for Securing Data in Fog Computing," *Scientific Reports*, vol. 15, art. 11654, Apr. 2025, doi: 10.1038/s41598-025-92245-9.

[15] S. B. N. Premakumari, G. Sundaram, M. Rivera, P. Wheeler, and R. E. Pérez Guzmán, "Reinforcement Q-Learning-Based Adaptive Encryption Model for Cyberthreat Mitigation in Wireless Sensor Networks," *Sensors*, vol. 25, no. 7, art. 2056, Mar. 2025, doi: 10.3390/s25072056.

[16] Sarkar S, Shafaei S, Jones TS, Totaro MW. Secure Communication in Drone Networks: A Comprehensive Survey of Lightweight Encryption and Key Management Techniques. *Drones*. 2025; 9(8):583. https://doi.org/10.3390/drones9080583.

[17] A. Zhaxygulova et al., "Secure and Energy-Aware Cryptographic Framework for IoT-Enabled UAV Systems" *Symmetry*, vol. 17, no. 11, art. 1987, 2025.

[18] Y. Nir and A. Langley, "ChaCha20 and Poly1305 for IETF Protocols," RFC 8439, Jun. 2018, doi: 10.17487/RFC8439.

[19] J. R. Rice, "The Algorithm Selection Problem," *Advances in Computers*, vol. 15, pp. 65–118, 1976, doi: 10.1016/S0065-2458(08)60520-3.

[20] K. A. Smith-Miles, "Cross-disciplinary Perspectives on Meta-learning for Algorithm Selection," *ACM Computing Studies*, vol. 41, no. 1, art. 6, 2009, doi: 10.1145/1456650.1456656.

[21] P. Brazdil, C. Giraud-Carrier, C. Soares, and R. Vilalta, *Metalearning: Applications to Data Mining*. Springer, 2009, doi: 10.1007/978-3-540-73263-1.

[22] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006, ISBN: 978-0-387-31073-2.

Поліщук Ю.Я., Проскурін Д.П., Гринюк Т.В.

Анотація. *Безпілотні літальні апарати (БПЛА) генерують гетерогенні потоки даних, зокрема відео в реальному часі, телеметрію та повідомлення керування і управління, функціонуючи за умов суворих обмежень пропускної здатності каналів зв'язку, енергоспоживання, обчислювальних ресурсів та рівня ризику виконання місії. Використання єдиного універсального алгоритму шифрування для всіх сценаріїв застосування БПЛА є неефективним, оскільки призводить або до надмірних обчислювальних і енергетичних витрат, або до недостатнього рівня криптографічного захисту у критичних умовах.*

У даній роботі формалізовано множину з восьми критеріїв K_1–K_8 для обґрунтованого вибору симетричних алгоритмів шифрування. На їх основі запропоновано підхід, що базується на методах машинного навчання (ML), який забезпечує автоматизований вибір найбільш релевантного криптоалгоритму відповідно до параметрів конкретного сценарію застосування БПЛА.

Ключовим науковим результатом є розроблення спеціалізованого датасету, орієнтованого на задачі БПЛА, який інтегрує параметри контексту місії (умови функціонування, обмеження ресурсів, рівень загроз) та криптографічні характеристики алгоритмів (стійкість, швидкодія, ресурсна ефективність). Запропонована структура ознак забезпечує формалізоване представлення задачі вибору алгоритму

та створює основу для навчання інтелектуальних моделей прийняття рішень.

Кожна група ознак обґрунтована сучасними науковими дослідженнями у галузях кібербезпеки БПЛА, легковагової криптографії та адаптивного шифрування із застосуванням ML. Робота також розвиває попередні результати авторів, пов'язані зі створенням датасету криптографічних алгоритмів та застосуванням нейронних мереж для забезпечення конфіденційності зображень у системах БПЛА.

***Ключові слова**: безпілотні літальні апарати (БПЛА), постквантова криптографія, машинне навчання, адаптивне шифрування, вибір криптографічного алгоритму.*

Поліщук Юлія Ярославівна, молодший науковий співробітник дослідник лабораторії протидії кіберзагрозам в авіаційній галузі, Державний університет «Київський авіаційний інститут».

Polischuk Yuliia, Junior Resea

E-mail: polishchuk.yu.ya@gmail.com

ORCID: 0000-0002-0686-2328

Проскурін Дмитро Петрович, к.т.н., науковий співробітник лабораторії протидії кіберзагрозам в авіаційній галузі, Державний університет «Київський авіаційний інститут».

Proskurin Dmytro, PhD, Researcher at Research Laboratory of Cyber Threats Counteraction in Aviation, State University «Kyiv Aviation Institute».

E-mail: dmytro.proskurin@kai.edu.ua

ORCID: 0000-0002-2835-4279

Гринюк Тетяна Василівна, молодший науковий співробітник лабораторії протидії кіберзагрозам в авіаційній галузі, Державний університет «Київський авіаційний інститут».

Hryniuk Tetiana, Junior Researcher at Research Laboratory of Cyber Threats Counteraction in Aviation, State University «Kyiv Aviation Institute».

E-mail: tetiana.hryniuk@kai.edu.ua

ORCID: 0009-0007-8717-6752