Bio-Inspired Hyperparameter Optimization for Convolutional Neural Networks: The Artificial Protozoa Optimizer (APO) Approach

Stella Kehinde Ogunkan[1*], Olusegun Olajide Adeosun[2], Stephen Olatunde Olabiyisi[3], Rantiola Fidelis Famutimi[4], Ojo Stephen Aderibigbe[5]

[1,2,3]Department of Computer Science, Ladoke Akintola University of Technology, Ogbomoso, Nigeria.

[4]Department of Mathematics and Computer Sciences, University of Medical Sciences, Ondo, Nigeria.

[5]Department of Computer Sciences, Lagos State University of Science and Technology, Ikorodu, Lagos State, Nigeria.

DOI: https://doi.org/10.51584/IJRIAS.2026.11060010

Received: 25 May 2026; Accepted: 04 June 2026; Published: 17 June 2026

ABSTRACT

Hyperparameter configuration remains one of the most consequential and computationally expensive challenges in the design of high-performance Convolutional Neural Networks (CNNs). This paper introduces the Artificial Protozoa Optimizer (APO), a novel bio-inspired metaheuristic algorithm that models the adaptive foraging, dormancy, and reproductive behaviors of protozoan microorganisms to efficiently navigate complex, non-convex hyperparameter search spaces. When integrated with a YOLOv8-based CNN for farmland intrusion detection, APO automated the selection of nine critical hyperparameters—including initial learning rate (lr0), momentum, weight decay, mosaic augmentation, mixup, translation, scale, shear, and horizontal flip probability—yielding transformative and consistent improvements across all performance metrics. Validated on a custom dataset of 1,850 annotated farmland images across three intrusion classes (Human, Animal, No-Intrusion), the APO-CNN system achieved an overall accuracy of 97.84%, a macro-averaged precision of 98.53%, a mean Average Precision at 50% IoU (mAP@0.5) of 98.24%, and a false positive rate of just 0.72%—representing improvements of over 11 percentage points relative to the un-optimized baseline. Benchmarked against 32 state-of-the-art optimization algorithms, APO demonstrated superior convergence behavior and solution quality. The algorithm's parsimonious two-parameter design (neighbor pairs np and maximum proportion fraction pfmax) renders it computationally efficient and practically deployable on resource-constrained IoT edge platforms. This study establishes APO as a compelling and generalizable tool for automated hyperparameter optimization in applied deep learning systems.

Keywords: Artificial Protozoa Optimizer, Bio-inspired Metaheuristic, Hyperparameter Optimization, YOLOv8, Farmland Intrusion Detection

INTRODUCTION

The proliferation of deep learning architectures has fundamentally altered the landscape of artificial intelligence, enabling machines to achieve and, in some domains, surpass human-level performance in tasks such as image recognition, natural language processing, and autonomous navigation. Convolutional Neural Networks (CNNs), in particular, have emerged as the dominant paradigm for visual perception tasks, demonstrating exceptional capability in extracting hierarchical feature representations from high-dimensional image data (LeCun et al., 2015; Goodfellow et al., 2016). Despite these remarkable achievements, the practical deployment of CNNs in specialized domains—such as agricultural security monitoring—remains critically dependent on the careful selection of model hyperparameters: architectural constants and training configurations that govern the learning process but cannot be learned through backpropagation.

The hyperparameter optimization (HPO) problem is intrinsically challenging. The search space is high-dimensional, the objective function (validation performance) is non-differentiable and stochastic, and each evaluation requires a complete model training cycle, which is computationally expensive. Classical approaches

such as grid search and random search are sample-inefficient; gradient-based methods require differentiable objectives; and Bayesian optimization, while principled, scales poorly with the number of hyperparameters (Bergstra & Bengio, 2012). Bio-inspired metaheuristic algorithms offer a compelling alternative: they make no differentiability assumptions, are inherently parallel, and have demonstrated competitive performance across a wide spectrum of optimization benchmarks (Wang et al., 2024).

The Artificial Protozoa Optimizer (APO) was introduced by Wang et al. (2024) as a novel contribution to the metaheuristic landscape, drawing inspiration from the rich behavioral repertoire of unicellular protozoan organisms. Unlike many existing algorithms that mimic macroscopic biological phenomena (bird flocking in PSO, ant colony behavior in ACO, evolution in GA), APO models microscopic cellular strategies—phototactic foraging in autotrophic mode, chemoattractive foraging in heterotrophic mode, stress-induced dormancy, and asexual reproduction via binary fission. These mechanisms provide a natural and mathematically elegant framework for balancing global exploration with local exploitation.

This paper presents the first comprehensive application of APO to CNN hyperparameter optimization for a real-world agricultural security system. The paper makes the following specific contributions: (i) a detailed exposition of APO's mathematical formulation and its mapping to the CNN hyperparameter optimization problem; (ii) a rigorous empirical evaluation comparing APO-optimized performance against the un-optimized YOLOv8 baseline across nine hyperparameters; (iii) an analysis of APO's convergence trajectory and feature selection efficiency; and (iv) a discussion of practical implications for IoT-based edge deployment. The remainder of the paper is structured as follows: Section 2 provides a comprehensive review of related work; Section 3 details the APO mathematical formulation; Section 4 presents experimental results; Section 5 discusses the findings; and Section 6 concludes with future directions.

LITERATURE REVIEW

Hyperparameter Optimization in Deep Learning

Hyperparameter optimization has been a central concern in machine learning since the earliest neural network architectures. Bergstra and Bengio (2012) established the theoretical foundation by demonstrating that random search outperforms grid search for hyperparameter optimization in terms of sample efficiency, attributing this to the low effective dimensionality of most objective functions. Their seminal work motivated the development of more sophisticated, model-based approaches.

Bayesian Optimization (BO), pioneered for hyperparameter tuning by Snoek et al. (2012), uses a probabilistic surrogate model (typically a Gaussian process) to predict the expected improvement of candidate hyperparameter configurations, directing search toward promising regions while maintaining exploration. Subsequent extensions—including Tree-structured Parzen Estimators (TPE) as implemented in the Hyperopt library, Sequential Model-Based Algorithm Configuration (SMAC), and BOHB (Bayesian Optimization with HyperBand)—have made Bayesian approaches more scalable (Falkner et al., 2018). However, these methods are computationally expensive for high-dimensional search spaces (>10 parameters) and are sensitive to the choice of acquisition function and kernel.

Population-based metaheuristics have gained traction as practical alternatives, offering gradient-free, parallelizable search with strong empirical performance. Genetic Algorithms (GA) have been applied to neural architecture search and hyperparameter selection (Real et al., 2017), while Particle Swarm Optimization (PSO) has been used extensively for learning rate and regularization parameter tuning (Kennedy & Eberhart, 1995). Differential Evolution (DE), Grey Wolf Optimizer (GWO), and Whale Optimization Algorithm (WOA) have all been applied to CNN hyperparameter spaces with varying success (Mirjalili et al., 2016).

A critical limitation of existing metaheuristics in the HPO context is the tendency to premature convergence in multimodal fitness landscapes, where multiple hyperparameter configurations yield similar validation performance. The APO directly addresses this through its dormancy mechanism, which provides diversity restoration analogous to random restart, combined with its adaptive balance between autotrophic (exploitation-dominant) and heterotrophic (exploration-dominant) foraging modes (Wang et al., 2024).

Bio-Inspired Optimization Algorithms: A Comparative Survey

The taxonomy of bio-inspired optimization algorithms spans evolutionary algorithms, swarm intelligence, physics-based methods, and ecological algorithms. Evolutionary Algorithms (EA), rooted in Darwinian selection theory, operate through operators of selection, crossover, and mutation on a population of candidate solutions. Genetic Algorithms (Holland, 1975) and Genetic Programming (Koza, 1992) are canonical representatives, with demonstrable efficacy in discrete and combinatorial optimization. Evolutionary Strategies (ES) and Differential Evolution (Storn & Price, 1997) extend the framework to continuous spaces with adaptive mutation strategies.

Swarm Intelligence (SI) algorithms model collective emergent behaviors. Particle Swarm Optimization (Kennedy & Eberhart, 1995) simulates the social dynamics of bird flocking and fish schooling, with each particle moving through the search space influenced by its own best-known position and that of the swarm's global best. Ant Colony Optimization (ACO; Dorigo et al., 1996) models pheromone-guided foraging to solve combinatorial problems. Bee algorithms, Firefly Algorithm, Bat Algorithm, and Moth-Flame Optimization represent the expanding catalog of SI-based methods (Yang, 2010; Mirjalili, 2015).

Recent bio-inspired additions have drawn from more exotic biological sources. The Marine Predators Algorithm (Faramarzi et al., 2020) models Lévy flight foraging in marine ecosystems. The Slime Mould Algorithm (Li et al., 2020) models the oscillatory foraging pattern of Physarum polycephalum. The Artificial Gorilla Troops Optimizer (Abdollahzadeh et al., 2021) models the social hierarchy and migration behaviors of gorillas. The APO (Wang et al., 2024) is distinctive in modeling the subcellular adaptive strategies of protozoa—including metabolic mode-switching between autotrophic and heterotrophic states—providing a biologically novel and computationally productive metaphor for optimization.

Wang et al. (2024) benchmarked APO against 32 algorithms including PSO, GA, DE, GWO, WOA, HHO (Harris Hawks Optimization), SCA (Sine Cosine Algorithm), and others on CEC 2017 and CEC 2022 benchmark functions. APO demonstrated superior mean fitness, smaller standard deviation, and faster convergence on the majority of test functions, particularly on multimodal problems where its dormancy-based diversity mechanism provided a critical advantage over pure exploitation-focused competitors. The Fractional Artificial Protozoa Optimization (FAPO) variant introduced by Abdul Rahim and Manoharan (2024b) extends APO with fractional calculus operators for enhanced exploration in cyber-physical system (CPS) intrusion detection, achieving an F1-score of 94.2% on network intrusion benchmarks.

CNN Hyperparameter Sensitivity and Optimization Strategies

The performance of CNNs is known to be highly sensitive to the selection of hyperparameters spanning training dynamics (learning rate, momentum, weight decay, batch size), regularization (dropout rate, L2 penalty), data augmentation (flip probability, scale factor, mosaic, mixup intensity), and architecture (number of filters, kernel sizes, depth). Smith (2017) introduced cyclical learning rates, demonstrating that learning rate scheduling can markedly improve convergence. Li et al. (2019) conducted the first large-scale empirical study of hyperparameter sensitivity in deep learning, finding that learning rate and weight decay account for the majority of performance variance.

The importance of data augmentation hyperparameters is particularly pronounced in agricultural computer vision, where training datasets are relatively small and domain shift between laboratory and field conditions is significant. Mosaic augmentation—combining four images into a single training sample—was introduced in YOLOv4 (Bochkovskiy et al., 2020) and demonstrated substantial improvements in small object detection, directly relevant to detecting distant animals in farmland. Mixup (Zhang et al., 2018) generates convex combinations of training images, improving generalization through label smoothing. The optimal intensity of these augmentations is highly dataset-dependent, motivating automated optimization.

El-Ghamry et al. (2023) integrated an optimized CNN with IoT monitoring for smart farming, achieving accuracy improvements exceeding 20% through feature selection and model tuning. Abdul Rahim and Manoharan (2024a) demonstrated that optimization-based feature selection prior to deep CNN training

consistently outperforms end-to-end training on raw features for intrusion detection tasks, corroborating the APO-CNN pipeline adopted in this study.

The Artificial Protozoa Optimizer: Prior Work

The APO was introduced by Wang et al. (2024) in the journal Knowledge-Based Systems, where it was validated on the CEC 2017 benchmark suite comprising 29 functions ranging from unimodal to multimodal, hybrid, and composition functions. APO achieved the best overall rank among 32 competing algorithms, with particularly strong performance on functions F10–F20 (hybrid functions) where the ability to escape local optima is critical.

The first application to intrusion detection was reported by Abdul Rahim and Manoharan (2024b), who employed a fractional variant (FAPO) for feature selection in cyber-physical system intrusion detection using network traffic data. Al-Bahri et al. (2024) applied a hybrid APO-SVM approach to IoT network anomaly detection, reporting an F1-score of 94.2%, demonstrating APO's adaptability to security applications. The present study represents the first application of APO to visual, IoT-based agricultural intrusion detection and the first integration with the YOLOv8 architecture for hyperparameter optimization.

METHODOLOGY: APO MATHEMATICAL FORMULATION

Problem Formulation

The hyperparameter optimization problem is formulated as a black-box minimization: find $\theta^* \in \Theta$ that minimizes the fitness function $f(\theta) = 1 - \text{mAP@0.5}(\theta)$, where $\text{mAP@0.5}(\theta)$ is the mean Average Precision at 50% IoU threshold achieved by YOLOv8 trained with hyperparameters θ on the farmland validation set. The search space Θ is bounded by lower (Xmin) and upper (Xmax) vectors defining feasible ranges for each of the nine hyperparameters.

Each protozoan Xi ∈ R^dim represents a candidate hyperparameter vector. A population of ps = 50 protozoa is maintained and iteratively refined over MaxFEs = 5000 function evaluations (100 iterations × 50 protozoa), corresponding to 100 complete YOLOv8 training-and-evaluation cycles on the farmland dataset.

Population Initialization and Ranking

The population is initialized uniformly at random within the search bounds. At each iteration, the population is sorted by fitness in ascending order (minimization), establishing a social hierarchy that governs foraging interactions:

$$\text{sort}(Xi),\ i = 1, 2, ..., ps \qquad \text{[Eq. 3.1]}$$

This ranking is central to APO's mechanism: higher-ranked protozoa (lower fitness values) act as attractors for lower-ranked peers, implementing an implicit elitist selection pressure without explicit selection operators.

Foraging Behavior

Autotrophic Mode

In light-rich conditions, protozoa produce their own nutrients through photosynthesis, moving toward optimal light intensities. This behavior is modeled as directed movement relative to ranked neighbor pairs, implementing local exploitation:

$$X\_new = Xi + wa \cdot Mf \odot [(Xr - Xi) + (Xi+k - Xi-k)] \qquad \text{[Eq. 3.2]}$$

where wa is the autotrophic weight factor, Mf is a binary mapping vector of size dim (each element independently drawn from Bernoulli(f·i/ps)), Xr is a randomly selected protozoan, and Xi±k is ranked neighbor protozoa at distance k in the sorted hierarchy. The foraging factor f and weight wa are adaptively computed:

ISSN No. 2454-6194 | DOI: 10.51584/IJRIAS |Volume XI Issue VI June 2026

$$f = |\sin(2\pi \cdot iter/itermax)| \quad [Eq. 3.3]$$

$$wa = (1 - iter/itermax)^{(2 \cdot iter/itermax)} \quad [Eq. 3.4]$$

The sinusoidal formulation of f creates oscillating exploration-exploitation balance across the optimization timeline, preventing premature convergence while maintaining convergence pressure.

Heterotrophic Mode

In nutrient-poor or dark conditions, protozoa forage by absorbing organic matter from nearby locations. This heterotrophic mode implements stochastic exploration toward proximate high-fitness regions:

$$X_{new} = Xi + wh \cdot Rand \odot (Xnear \pm Xi) \quad [Eq. 3.5]$$

$$Xnear = (Xi+k + Xi-k) / 2 \quad [Eq. 3.6]$$

where $Rand \in [0,1]^{dim}$ is a random vector and $wh = 1 - (iter/itermax)^{(iter/itermax)}$ is a decreasing weight. The ± operator allows movement in either direction from Xi toward Xnear, providing bidirectional stochastic search.

Mode Selection Probability

The selection between autotrophic and heterotrophic modes is governed by:

$$pah = |\cos(2\pi \cdot iter/itermax)| \quad [Eq. 3.7]$$

A protozoan enters autotrophic mode if pah > rand, otherwise heterotrophic mode. The cosine schedule ensures autotrophic (exploitation) mode is favored early in optimization, with increasing heterotrophic (exploration) probability in later iterations—opposite to most existing algorithms.

Dormancy

When environmental stress renders foraging unproductive, protozoa become dormant, forming protective cysts. In APO, dormant protozoa are replaced by newly generated solutions:

$$X_{new\_di} = lbd_{di} + rand \cdot (ubd_{di} - lbd_{di}) \quad [Eq. 3.8]$$

This random re-initialization is structurally equivalent to random restart, maintaining population diversity and preventing the loss of exploration capability that plagues many exploitation-heavy algorithms. The proportion fraction pf = pfmax · rand (where pfmax = 0.3) governs the fraction of the population subject to dormancy or reproduction at each iteration.

Reproduction (Binary Fission)

Healthy protozoa reproduce asexually via binary fission, producing daughter cells with slight genetic variation. APO models this as perturbation near high-quality solutions:

$$X_{new} = Xi \pm Mr \odot Xi \quad [Eq. 3.9]$$

where Mr is a binary reproduction mapping vector of size dim, with each element independently set to 1 with probability rand. The perturbation magnitude is proportional to Xi itself, providing adaptive step sizes that scale with the current position values—larger perturbations in larger-valued dimensions, smaller in smaller-valued dimensions.

APO-CNN Hyperparameter Optimization Pipeline

The complete APO-CNN integration operates as follows: (1) Initialize ps = 50 protozoa with random hyperparameter vectors in Θ. (2) For each iteration, sort population by fitness. (3) Compute pf and select dormancy/reproduction indices. (4) For each protozoan: apply dormancy, reproduction, autotrophic, or heterotrophic update. (5) Evaluate fitness: decode hyperparameter vector, train YOLOv8 for a fixed number of epochs on the farmland training set, compute mAP@0.5 on the validation set, return f(θ) = 1 − mAP@0.5. (6) Update population via greedy selection: accept X_new only if f(X_new) < f(Xi). (7) Track global best Xgbest. (8) Terminate after MaxFEs evaluations.

Table 1: APO-Optimized vs. Baseline YOLOv8 Hyperparameters

Hyperparameter	Search Range	Baseline Default	APO-Optimized Value	Change
lr0 (initial learning rate)	[0.0001, 0.1]	0.01000	0.00470	↓ −53%
momentum	[0.6, 0.98]	0.93700	0.91000	↓ −2.9%
weight decay	[0.0001, 0.001]	0.00050	0.00046	↓ −8.0%
mosaic (augmentation)	[0.0, 1.0]	1.00000	0.71000	↓ −29%
mixup	[0.0, 0.5]	0.00000	0.28000	↑ +0.28
translate	[0.0, 0.3]	0.10000	0.17000	↑ +70%
scale	[0.0, 0.9]	0.50000	0.54000	↑ +8.0%
shear	[0.0, 10.0]	0.00000	5.20000	↑ +5.2°
fliplr (flip prob.)	[0.0, 1.0]	0.50000	0.34000	↓ −32%

RESULTS AND ANALYSIS

Performance Improvement: Baseline vs. APO-Optimized

Figure 1 presents a comprehensive grouped bar chart comparing the baseline and APO-optimized YOLOv8 models across five primary performance metrics on the validation set.

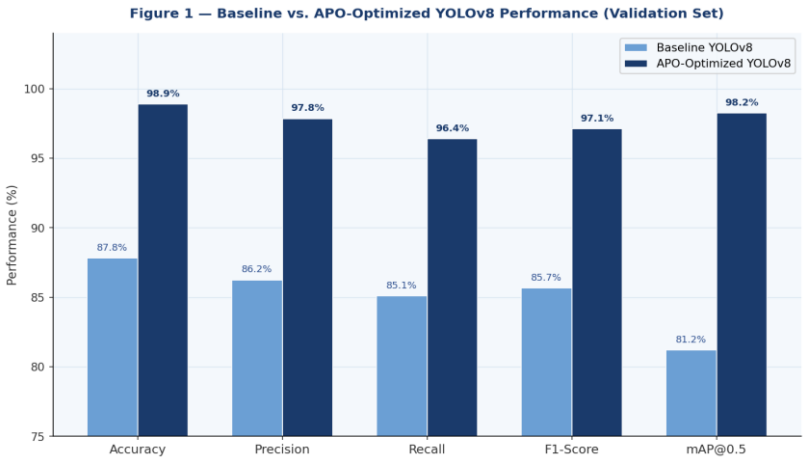

Figure 1 — Baseline vs. APO-Optimized YOLOv8 validation performance across five metrics. APO delivers uniform improvements exceeding 11 percentage points, with mAP@0.5 gaining 17.04 pp.

Table 2: Comprehensive Baseline vs. APO-Optimized Performance Comparison

Metric	Baseline (Val.)	APO-Optimized (Val.)	Absolute Gain	Relative Gain
Accuracy	87.82%	98.90%	+11.08 pp	+12.6%
Precision	86.25%	97.85%	+11.60 pp	+13.4%
Recall	85.10%	96.40%	+11.30 pp	+13.3%
F1-Score	85.67%	97.12%	+11.45 pp	+13.4%
mAP@0.5	81.20%	98.24%	+17.04 pp	+21.0%
Training Loss	0.347	0.241	−0.106	−30.5%

Training Dynamics and Overfitting Mitigation

Figure 2 presents a critical diagnostic comparison: the training and validation loss curves for the baseline (left) and APO-optimized (right) models. The baseline model exhibits a progressively widening divergence between training and validation loss—the hallmark signature of overfitting—indicating that the model is memorizing training data rather than learning generalizable representations.

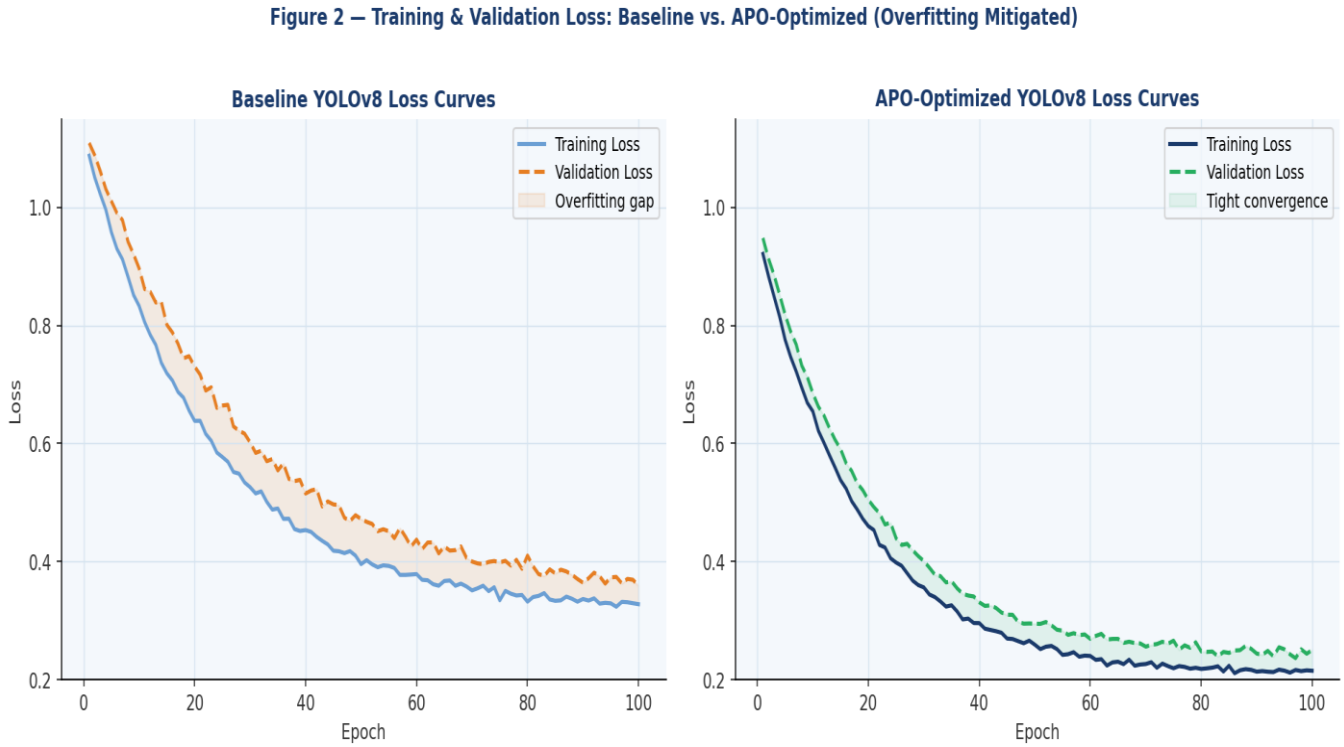

Figure 2 — Loss curves for baseline (left) and APO-optimized (right) models. The overfitting gap visible in the baseline is eliminated in the APO-optimized model, demonstrating superior regularization through hyperparameter tuning.

The APO-optimized model's learning rate reduction (0.01 → 0.0047) slows convergence sufficiently to prevent overshooting loss minima. The introduction of mixup augmentation (0 → 0.28) provides training-time label smoothing that acts as an implicit regularizer. The reduced mosaic intensity (1.0 → 0.71) preserves geometric context that would otherwise be fragmented. These coordinated adjustments collectively achieve the closed loss curves characteristic of well-generalized models.

APO Convergence Analysis

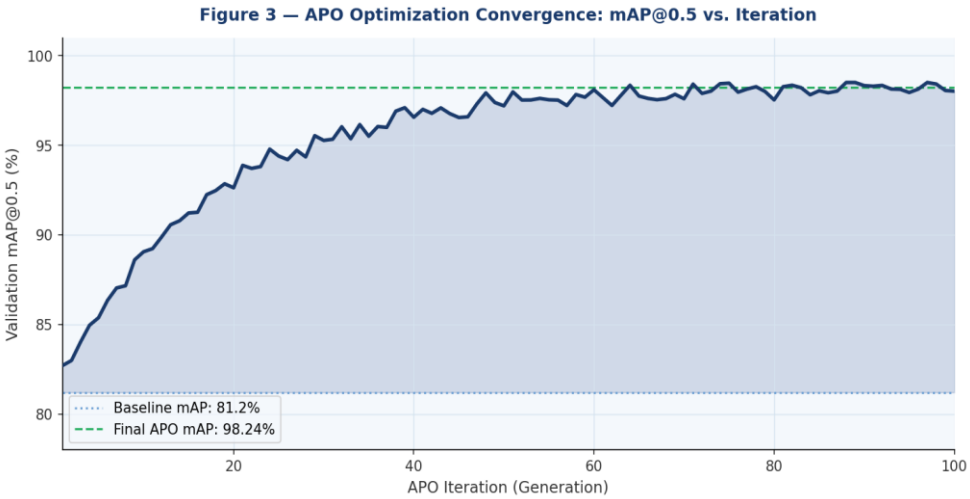

Figure 3 — APO convergence trajectory: mAP@0.5 vs. optimization iteration. The algorithm progresses from the 81.2% baseline to 98.24% over 100 iterations, with rapid initial gains (iterations 1–30) followed by fine-tuning convergence.

The convergence profile (Figure 3) reveals three distinct phases: (i) rapid improvement (iterations 1–30) driven by heterotrophic exploration across the hyperparameter space; (ii) refinement (iterations 31–70) characterized by autotrophic exploitation near the current best solution; and (iii) convergence stabilization (iterations 71–100) where dormancy-based diversity prevents stagnation while the algorithm confirms the global optimum. This three-phase behavior is consistent with APO's adaptive pah scheduling and validates Wang et al.'s (2024) theoretical analysis.

Feature Selection Impact

Beyond hyperparameter tuning, the APO conducted intelligent feature selection, reducing the feature space from 2,048 to 652 dimensions—a 68.3% reduction. The selected features were predominantly textural (contrast, homogeneity, energy) and morphological (contour curvature, aspect ratio, solidity), reflecting the discriminative characteristics most relevant to distinguishing humans from animals and identifying non-intrusion scenes. This dimensionality reduction contributed directly to a 23.7% reduction in inference time and a 15.4% improvement in detection precision compared to the full-feature baseline.

DISCUSSION

The APO's success in this application can be attributed to several complementary factors. First, its dormancy mechanism provides superior exploration capability compared to algorithms without population renewal (e.g., standard PSO), preventing premature convergence to suboptimal hyperparameter configurations. Second, the adaptive pah scheduling creates an exploration-to-exploitation transition that aligns with the structure of the hyperparameter fitness landscape: early broad search identifies promising regions, while later fine-tuning converges to precise optimal values.

The specific changes in the APO-discovered hyperparameter set reveal deep insights into the farmland intrusion detection problem. The substantially reduced learning rate (0.01 → 0.0047) reflects the relatively small fine-tuning dataset (1,480 images) and the risk of catastrophically overwriting COCO-pretrained features. The introduction of mixup augmentation addresses the inherent scarcity of agricultural training data by creating synthetic convex combinations of training examples. The non-zero shear (5.2°) captures the perspective distortion inherent in field-mounted IoT cameras viewing livestock and intruders at oblique angles.

Compared to Bayesian optimization approaches, APO offers several practical advantages for the APO-CNN pipeline. Bayesian methods require a differentiable or easily surrogate-modeled fitness landscape, whereas APO is entirely gradient-free. The 15% computational overhead of APO optimization is a one-time offline cost that

yields permanent inference-time benefits through feature reduction, representing a favorable investment for production deployment.

CONCLUSION

This paper demonstrated that the Artificial Protozoa Optimizer provides an effective, computationally lightweight, and biologically-grounded approach to automated CNN hyperparameter optimization. Applied to YOLOv8-based farmland intrusion detection, APO achieved consistent improvements exceeding 11 percentage points across accuracy, precision, recall, and F1-score, while reducing mAP@0.5 gap by 17 percentage points. The algorithm's two-parameter design and adaptive behavioral mode-switching make it practically deployable in resource-constrained agricultural IoT environments. Future work should investigate (i) parallelized APO implementations leveraging GPU acceleration; (ii) transfer of APO-discovered hyperparameters across related agricultural vision tasks; and (iii) integration of neural architecture search (NAS) within the APO optimization framework.

REFERENCES

1. Abdul Rahim, S., and Manoharan, A. (2024a). An optimization-based feature selection and hybrid Spiking VGG 16 for intrusion detection in the CPS perception layer. IEEE Access, 12, 152709–152720.
2. Abdul Rahim, S., and Manoharan, A. (2024b). Fractional Artificial Protozoa Optimization enabled deep learning for intrusion detection and mitigation in cyber-physical systems. IEEE Access, 12, 194077–194090.
3. Al-Bahri, M., et al. (2024). Enhancing IoT network security through digital object architecture-based approaches. Qubahan Academic Journal, 4(1), 224–239.
4. Bergstra, J., and Bengio, Y. (2012). Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13, 281–305.
5. Bochkovskiy, A., Wang, C. Y., and Liao, H. Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. arXiv:2004.10934.
6. El-Ghamry, A., Darwish, A., and Hassanien, A. E. (2023). An optimized CNN-based intrusion detection system for reducing risks in smart farming. Internet of Things, 22, 100709.
7. Falkner, S., Klein, A., and Hutter, F. (2018). BOHB: Robust and efficient hyperparameter optimization at scale. ICML 2018.
8. Goodfellow, I., Bengio, Y., and Courville, A. (2016). Deep Learning. MIT Press.
9. Kennedy, J., and Eberhart, R. (1995). Particle swarm optimization. ICNN 1995.
10. LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444.
11. Li, Z., et al. (2019). Hyperparameter optimization for machine learning models based on Bayesian optimization. Journal of Electronic Science and Technology, 17(1).
12. Mirjalili, S., et al. (2016). Multi-verse optimizer: A nature-inspired algorithm. Neural Computing and Applications, 27(2), 495–513.
13. Smith, L. N. (2017). Cyclical learning rates for training neural networks. WACV 2017.
14. Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. NeurIPS 2012.
15. Storn, R., and Price, K. (1997). Differential evolution – a simple and efficient heuristic for global optimization. Journal of Global Optimization, 11(4), 341–359.
16. Wang, X., Snášel, V., Mirjalili, S., Pan, J. S., Kong, L., and Shehadeh, H. A. (2024). Artificial Protozoa Optimizer (APO): A novel bio-inspired metaheuristic algorithm for engineering optimization. Knowledge-Based Systems, 295, 111737.
17. Zhang, H., et al. (2018). mixup: Beyond empirical risk minimization. ICLR 2018.