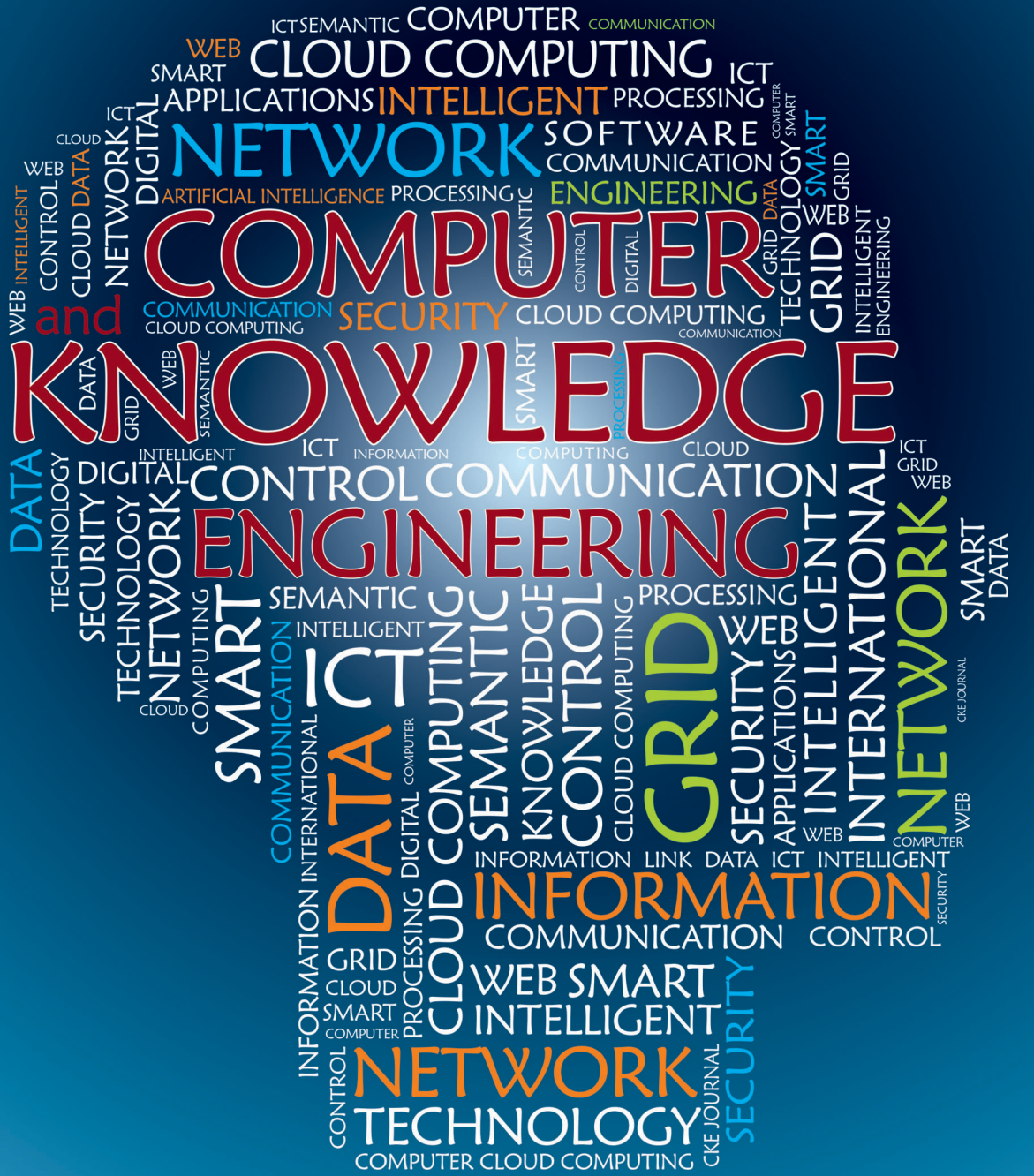
FACULTY OF ENGINEERING

Journal of Computer and Knowledge Engineering

Vol. 8, No. 2, 2025 ISSN: 2717-4123



Serial No. 16
Autumn & Winter 2025

Journal of Computer & Knowledge Engineering
Vol. 8, No.2, 2025
Serial No. : 16

Journal of
COMPUTER AND KNOWLEDGE ENGINEERING

Ferdowsi University of Mashhad

ISSN: 2717-4123

Journal of Computer and Knowledge Engineering
Faculty of Engineering, Ferdowsi University of Mashhad
P. O. Box. 91775-1111, Mashhad, I.R. IRAN
Tel: +98 51 38806024, Fax: +98 51 38763301, Email: cke@um.ac.ir, Site: cke.um.ac.ir

CONTENTS

Journal of Computer and Knowledge Engineering, Vol. 8, No. 2, 2025. (1-10)

Ferdowsi University of Mashhad

Journal of Computer and Knowledge Engineering

https://cke.um.ac.ir

Information and Communication Technology Association of Iran

Advancing Over-the-Air Federated Learning through Deep Reinforcement Learning in UAV-Assisted Networks with Movable Antennas*

Research Article

Mohsen Ahmadzadeh[1], Saeid Pakravan [2], Ghosheh Abed Hodtani [3]

10.22067/cke.2025.91290.1139

Abstract *This paper investigates the deployment of over-the-air federated learning (OTA-FL), leveraging the dynamic repositioning and line-of-sight communication capabilities of unmanned aerial vehicles (UAVs) and movable antennas to enhance network efficiency. A closed-form expression is derived to quantify the optimality gap between the actual federated learning (FL) model and its theoretical ideal, accounting for the capabilities of movable antennas to show the diverse relationship between Mean Square Error (MSE) and the optimality gap. Then An MSE minimization problem is then formulated, involving the joint optimization of moveable antenna position vectors, and the beamforming vector at the UAV. This complex non-convex problem is reformulated as a Markov Decision Process (MDP) and solved using the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm within the deep reinforcement learning (DRL) framework. Numerical results demonstrate that the proposed algorithm outperforms benchmarks such as Advantage Actor-Critic(A2C) and Soft Actor-Critic (SAC).*

Key Words Over-the-air federated learning, Deep reinforcement learning, Unmanned aerial vehicles, Movable Antenna.

1. INTRODUCTION

Federated Learning (FL) is a secure method for collaboratively building a unified model across various participants. Yet, its application in practice often encounters difficulties caused by restricted data exchange capabilities [1] [2]. To tackle the challenge of achieving minimal delay and broad connectivity in IoT-driven Federated Learning, an innovative solution known as over-the-air FL (OTA-FL) has been developed [3] [4]. This technique maximizes the efficient use of bandwidth by leveraging the inherent combining feature of wireless access networks through analog signaling. OTA-FL achieves model integration by utilizing the overlapping characteristics of wireless signals, where updates from edge devices are merged into a collective representation. This process employs over the air computation (AirComp) to perform direct aggregation, avoiding the step of separately processing each parameter. Consequently, it decreases delays and boosts resource efficiency by operating within common time and frequency allocations.

Unmanned aerial vehicles (UAVs) are increasingly playing a pivotal role in modern wireless communication systems, owing to their cost-effectiveness, high mobility, and versatile capabilities. These vehicles are capable of operating as aerial base stations, relays, or access points, which significantly extend coverage and ensure reliable line-of-sight (LoS) connectivity for data transmission across various environments [5].

The evolution of communication systems has led to the widespread adoption of Multiple-Input Multiple-Output (MIMO) technology, characterized by the use of multiple antennas [6, 7]. MIMO systems are primarily designed to improve channel capacity, boost data transmission rates, and optimize various performance parameters of communication networks. [8, 9]. Traditional fixed-position antennas (FPAs) often face limitations in achieving optimal beamforming gains within dynamic environments. To address this limitation, we propose integrating movable antennas (MAs) into OTA-FL systems, allowing for real-time adaptation to varying wireless channel conditions [4, 10]. Leveraging MAs at the receiving server enhances OTA-FL performance by utilizing spatial degrees of freedom (DoF). Unlike FPAs, MAs offer the capability to reconfigure the wireless environment dynamically, introducing extra DoFs that substantially boost the efficiency and effectiveness of

* Manuscript received 2024 December 23, Revised 2025 March 4, Accepted 2025 May 13.

[1]Phd Candidate, Department of Electric and Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran.

[2]Phd Candidate, Department of Electric and Computer Engineering, Laval University, Quebec City, Canada

[3]Corresponding Author. Professor, Department of Electric and Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran. **Email:** hodtani@um.ac.ir

OTA-FL systems [4, 11].

Reinforcement Learning (RL) is a promising approach for autonomous decision-making, where an agent learns by interacting with its environment, taking actions, and adjusting its strategy based on rewards to optimize performance [12]. However, RL struggles with large-scale environments due to its high demand for computational resources. To address this, Deep Reinforcement Learning (DRL) leverages deep neural networks, enabling more efficient learning in complex, high-dimensional environments. DRL methods are particularly useful for modern networks with high computational complexity. Additionally, while centralized RL can create significant signalling overhead, DRL allows for decentralized multi-agent systems, where agents make independent decisions, reducing overhead and improving scalability, especially in applications like UAV-assisted networks [13, 14].

This study introduces a UAV-assisted MA-assisted framework that OTA-FL. The main contributions of this paper are outlined as follows:

1. System Design: This paper introduces a UAV-enabled MA-architecture that incorporates OTA-FL in the AP.
2. Optimality Gap Analysis: We perform a comprehensive optimality gap evaluation, deriving a closed-form expression to quantify the gap between the achieved and optimal loss. This analysis reveals how MSE impact the convergence behaviour of the OTA-FL algorithm.
3. Performance Assessment: We conduct extensive simulations to evaluate the effectiveness of the learning network. The results show that the TD3 method surpasses all single-agent techniques, including Advantage Actor-Critic(A2C) and Soft Actor-Critic (SAC).

1.1. Related work

Numerous research efforts have explored the use of UAVs and MA in OTA-FL. In the following section, we present an in-depth analysis of these studies.

To improve the efficiency of OTA-FL, various research efforts have utilized UAV to address challenges related to magnitude alignment during model aggregation at the edge server. In [15], the Fog-aided Internet of Drones framework employs machine learning to analyse data collected by drones at fog nodes, offering various services. FL is utilized to enhance data privacy by enabling local drone training and sharing model parameters instead of raw data. However, privacy risks persist due to potential eavesdropping on uploaded parameters. The study focuses on optimizing drone power control to maximize the FL system's security rate while meeting battery and quality of service (QoS) constraints. A non-linear programming approach is proposed, and simulations validate the algorithm's effectiveness. In [16], a federated learning-based framework, Aerial Edge, is proposed for orchestrating aerial edge computing systems using UAVs. The approach employs multi-output regression to optimize resource allocation and execution time, selecting UAVs with suitable resources and flight time. A bin-packing optimization variant is introduced for efficient task scheduling, achieving fast execution and improved resource utilization, validated with real-world data. In [17], UAV swarms leveraging FL are studied to enable edge intelligence while addressing bandwidth and energy limitations. To minimize training energy consumption, the study jointly optimizes convergence thresholds, iterations, resource, and bandwidth allocation under accuracy and latency constraints. A fairness-focused variant minimizes maximum energy consumption across UAVs. Simulations demonstrate superior energy efficiency compared to baseline approaches. In [18], UAVs are employed in CR networks to leverage their high mobility and LoS transmission. However, spectrum sharing can cause interference, reducing the throughput of secondary users. RIS are utilized to mitigate this interference by reconstructing propagation links. The study focuses on maximizing SU throughput while ensuring primary user interference constraints are met, through joint optimization of UAV trajectory, RIS passive beamforming, and UAV power allocation. The problem is divided into three subproblems: beamforming, power allocation, and trajectory design, and an alternating iterative optimization algorithm is proposed. Numerical results demonstrate significant throughput improvement. In [19], a joint subchannel assignment and power allocation algorithm is proposed for NOMA-enabled cognitive satellite-UAV-terrestrial networks to optimize the sum rate of the secondary network under imperfect channel state information. The problem, constrained by interference temperature for primary users, minimum secondary user rates, UAV power limits, and subchannel capacity, is formulated as a mixed-integer non-linear programming task. It is addressed by decoupling into subchannel assignment and power allocation subproblems, solved using heuristic and successive convex approximation methods, respectively. Simulations demonstrate the algorithm’s superior performance in large-scale networks compared to benchmarks. In [4], the authors study an OTA-FL system with MAs at the AP to enhance

learning performance. They derive the optimality gap to evaluate FA mobility's impact and propose a nonconvex optimization framework to jointly optimize FA positions and beamforming. The problem is modeled as a MDP and solved using the recurrent deep deterministic policy gradient algorithm. Simulations show the FA-assisted OTA-FL system outperforms fixed-antenna systems, with RDPG surpassing existing methods. In [20], we explore the application of DRL techniques to design MA for OTA-FL in UAV networks, aiming to enhance the overall network performance by optimizing the antenna positions. By considering UAVs as FL clients, we demonstrate the efficacy of this approach in improving the communication and learning capabilities within the network. In [21], the authors propose an OTA-FL framework using movable antennas (MAs) and UAVs for IoT support. They minimize MSE via joint antenna and beamforming optimization, modeled as an MDP and solved with TD3. Simulations show TD3-based MA systems outperform FPA and other DRL methods, achieving higher rewards and better performance.

1.2. Organization

This paper is structured as follows: Section 0provides a detailed explanation of the system architecture, covering OTA-FL techniques and the UAV-enabled communication framework. In Section 0, the convergence behavior of the OTA-FL method is analyzed. Section 0 formulates the optimization problem, with a focus on the optimality gap. Section 0 presents a DRL-based framework for optimization. The simulation setup, experimental scenarios, and comparative results with existing benchmarks are discussed in Section 0 to validate the proposed approach. The paper concludes with Section 0

2. SYSTEM MODEL

We focus on the upload phase of an OTA-FL framework, which involves$N$single-antenna user equipment (UE) devices denoted as $UE_n, \forall n = [1, \dots, N]$, referred to as FL clients. These clients are randomly distributed across a designated area to collect local datasets, train local models, and collaboratively optimize a global model. The training of the global model is coordinated by a UAV equipped with K movable antennas (MA-UAV). The UAV moves randomly within the area of interest to facilitate communication and coordination with the FL clients.

We analyze the OTA-FL framework, where full participation involves performing sequential tasks in each training round. The process of OTAFL is as follows: The UAV transmits the updated global model, $v_t \in \mathrm{R}^q$ to all UEs, with q representing the size of the model parameter space. Each UE_nupdates its local model using the gradient descent method, described as:

$$v_{n,t} = v_t - \gamma \nabla G(v_t, S_n), \tag{1}$$

here, γ represents the learning rate, $\nabla G(v_t, S_n)$ denotes the gradient of the local loss function, and S_n signifies the local dataset for UE_n , $|S_n| = S$, with $|S_n|$ indicating its size. Each UE sends its updated local model back to the UAV, which aggregates these models by averaging them to update the global model, expressed as:

$$v_{t+1} = \frac{1}{N} \sum_{n=1}^{N} v_{n,t}. \tag{2}$$

This process is repeated iteratively until the predefined maximum number of outer iterations is achieved.

The UAV is equipped with a K MAs, which can be adjusted along a one-dimensional segment of length D with $[0, D]$. Each MA's position is restricted to the interval $[0, D]$ maintaining a minimum spacing of D_0 between adjacent antennas to prevent coupling. The positions of the K MAs are represented by the vector $d = [d_1, \dots, d_K]$, with their movement confined to a single dimension as defined by $d_1 < d_2 \dots < d_K$.

Under the assumption of line-of-sight (LoS) propagation conditions, the channel between the n-th UE and the UAV denoted as $g_n[d] \in C^{K\times 1}$, is expressed as:

$$g_n[d] = \sqrt{\frac{\ell_0}{{x_n}^{\alpha}}} [e^{j\frac{2\pi}{\lambda} d_1 \cos(\theta_n)}, \dots, e^{j\frac{2\pi}{\lambda} d_K \cos(\theta_n)}]^T, \tag{3}$$

here, ℓ_0 represents the path loss at the reference distance, λ denotes the wavelength, and α is the path loss exponent. Additionally, x_n and θ_n correspond to the distance between the MAs and the n-th UE, and the angle of arrival (AoA) of the LoS path, respectively. These values are determined based on the UAV locations during each training round.

In this context, it is assumed that UAV operates within a predefined area and transmits global model parameters from a fixed position. Moreover, because the signal path length is substantially greater than the extent of MA movement, the MA field condition between the UAV and UEs is presumed to hold. Consequently, θ_n and $x_n$are treated as constants during the transmission phase.

During the t-th training round, the UAV receives the local model parameters from all UEs, expressed as:

$$y = \sum_{n=1}^{N} p_n g_n[d] v_n + z, \tag{4}$$

here, p_n represents the transmission power factor for the n-th UE, and $z \in \mathrm{C}^{q\times N}$ denotes an additive white Gaussian noise (AWGN) matrix, where each element follows a complex normal distribution $\mathrm{CN}(0, \sigma^2)$. The aggregated model parameter vector $\hat{v}$ in the t-th training round is obtained by applying post-processing to the received signal at the UAV expressed as:

$$\begin{aligned} \hat{v}_{t+1} &= \frac{1}{N} \left(\frac{1}{\sqrt{\eta}} W^H y\right) \\ &= \frac{1}{N} \sum_{n=1}^{N} \frac{1}{\sqrt{\eta}} W^H p_n g_n[d]\, v_n + \frac{w^H z}{N\sqrt{\eta}}, \end{aligned} \tag{5}$$

here, $W \in C^{N\times 1}$ represents the beamforming vector at the UAV, and η denotes the scaling factor used for aligning the signal amplitude.

As outlined in [4], maximum power factor in each UEs should satisfies:

$$\frac{1}{q} {p_n}^2 E[|v_n|^2] \le P, \qquad 1,\dots,N. \tag{6}$$

It is assumed that the UAV starts and concludes the FL process at the same location, with its maximum allowable speed represented as V_{max}in meters per second (m/s). The UAV's movement is subject to the following constraints:

$$\begin{aligned} &|l[t+1] - l[t]|^2 \le V_{max} \\ &l[0] = [0,0,0], \\ &l[T] = [0,0,0], \end{aligned} \tag{7}$$

here, $l[t]$ represents the location of the UAV at time slot

t, and δdenotes the flying time between two consecutive time slots. In order to make easy to read and follow equation in Table 1. is summarized all character.

TABLE 1
summaries of all parameters

parameter	Definition
N	The total number of FL clients.
K	The number of MA on the UAV.
v_t	The global model at timeslot t
γ	Denotes the learning rate parameter.
q	The dimensionality of the model parameter space.
$\nabla G(v_t, S_n)$	Denotes the gradient of the local loss function.
S_n	signifies the local dataset for n-th UE
D	Refers to a one-dimensional segment of the MA antenna length.
D_0	The minimum spacing between adjacent antennas to prevent coupling.
$g_n[d]$	Represents the wireless channel between the n-th user equipment and the UAV.
ℓ_0	Denotes the path loss at the reference distance.
λ	Denotes the wavelength.
α	Represents the path loss exponent.
x_n	Denotes the distance between the MA antennas and the n-th UE.
θ_n	the AoA of the LoS path.
p_n	Represents the transmission power factor for the n-th UE.
z	Denotes an AWGN matrix, where each element follows a complex normal distribution.
W	Represents the beamforming vector at the UAV.
η	Denotes the scaling factor used for aligning signal amplitude.
V_{max}	Represents the speed of the UAV in meters per second (m/s).
$l[t]$	ŧepresents the location of the UAV at timeslot t
δ	Denotes the flying time between two consecutive timeslots.
r	Parameter representing the assumption of smoothness of model
μ	Parameter related to the PL inequality.
Λ	Denotes the upper limit of the model parameter.
S_t	Represents the states of the wireless environment for the MDP problem.
a_t	Denotes the action space of the MDP agent.
$reward_t$	Represents the reward function in the MDP framework.
π_φ	Denotes the policy function in DRL.
ϑ_1, ϑ_2	Represents the parameters of the critic network in DRL.

3. CONVERGENCE ANALYSIS

To support our convergence analysis, we adopt the following widely accepted assumptions as outlined in [4], [17]: the global loss function is r-smooth, meaning that for any given model parameters, there exists a nonnegative constant r for any given model parameters $v_1, v_2 \in R^q$, such that:

$$G(v_1) - G(v_2) \leq (v_1 - v_2)^T \nabla G(v_1) + \frac{r}{2}||v_1 - v_2||^2. \quad (8)$$

Where r is a measure of how smooth the function is, a smaller value indicates a smoother function. Additionally, the loss function satisfies the Polyak–Łojasiewicz (PL) inequality, ensuring that:

$$|G(v)|^2 \geq 2\mu[G(v) - G(v^*)] \quad (9)$$

Where, if $G(v^*)$ represents the optimal global loss value and where$\mu \geq 0$ is the PL constant. Lastly, the model parameters for each UE are bounded by an upper limit, ensuring that for $\Lambda \geq 0$we have:

$$E[|v|^2] \leq \Lambda \quad (10)$$

Theorem 1: Under the conditions specified in above assumptions, and by setting the learning rate $\frac{1}{r}$, the optimality gap after Trounds of training is bounded by:

$$E[G(v_{T+1}) - G(v^*)] \leq (1 - \frac{\mu}{r})^T (E[G(v_1)] - E[G(v^*)]) + \sum_{t=1}^{T} (1 - \frac{\mu}{r})^{T-t} MSE_t. \quad (11)$$

To further refine the bounds, MSE_t can be expressed as follows:

$$MSE_t = \frac{r\sigma^2\Lambda}{2N^2 P_{max} \max\limits_{n\in[1,\dots,N]} \frac{|w_t|^2}{|w_t g_t[d]|^2}} \quad (12)$$

Proof: See Appendix.

4. PROBLEM FORMULATION

According to Theorem 1, the optimality gap, which reflects the learning performance of OTAFL, can be expressed in terms of the MSE in each communication round. This is determined based on the relationship between the model updates, the aggregation error introduced by wireless communication, and the cumulative effect of these errors over multiple rounds. To enhance the learning performance, we can formulate an optimization problem aimed at jointly optimizing key system parameters, such as $\mathrm{w} = [w_1, \dots \mathrm{w}_N]^Tandd = [d_1, \dots, d_N]$, with the objective of minimizing the MSE as follow:

$$\min_{d,w} \ \mathrm{MSE}_t$$
$$\text{S.t.} \quad C_1: 0 \leq d_n \leq D \qquad \mathrm{n} \in [1,\dots,\mathrm{N}],$$
$$C_2: d_1 \leq d_2 \dots \leq d_N,$$
$$C_3: d_n - d_{n-1} \geq D_0 \quad n \in [1,\dots,\mathrm{N}],$$

In this context, several constraints are defined to

regulate the behavior of the MAs. These constraints include limiting the valid range within which the positions of the MAs can be located in C_1, determining the sequence in which the MAs are positioned in C_2 and enforcing a minimum required distance between neighboring Mas in C_3. The complexity of the problem is further heightened by the inherent nonconvexity present in the objective function, categorizing it as a nonconvex optimization problem.

Traditional mathematical optimization techniques, commonly referenced in the literature, encounter significant difficulties when applied to such problems. This is primarily due to the high dimensionality of the optimization variables and the dynamic, unpredictable nature of the underlying system characteristics. To address these challenges effectively, we propose leveraging a DRL algorithm, which offers greater flexibility and adaptability to accommodate the varying configurations and demands of the system.

5. PROPOSED DRL ALGORITHM

In this segment, we begin by reinterpreting the optimization challenge as a MDP, laying the foundation for addressing it using the TD3 algorithm. To tackle this problem, a DRL agent is implemented at the UAV, aiming to develop an optimal decision-making strategy that significantly boosts the efficiency of OTA-FL. The proposed framework leverages the MDP structure, enabling systematic problem-solving. The specifics of the MDP's state representation, action space, and reward mechanism are elaborated below for clarity and completeness.

State Space: The state space at time slot t captures the key environmental and system parameters that influence decision-making. It includes the distances between the MAs and clients, as well as the AoA for the LoS paths associated with these entities. These parameters collectively describe the dynamic state of the system at time t. Mathematically, the state space is represented as:

$$S_t = [[x_1, \ldots, x_n], [\theta_1, \ldots, \theta_n]]. \quad (13)$$

Action Space: The action space at a given time slot t defines the set of controllable variables that the agent can adjust to optimize system performance. It encompasses two critical components: the beamforming vector, which directs the transmitted signal's power and phase, and the spatial locations of the MAs. These actions collectively determine the system's ability to adapt to environmental changes and maximize efficiency. Mathematically, the action space at time slot t is represented as:

$$a_t = [[d_1, \ldots, d_N], [w, \ldots, w_N]]. \quad (14)$$

These variables collectively constitute the optimization parameters that require iterative refinement at each time step.

Reward Function: The reward function is designed to align with the optimization objective while adhering to system constraints. It evaluates the agent's actions by penalizing deviations from the desired outcome, ensuring that the learning process encourages improvements in system performance. The reward is mathematically defined as:

$$reward_t = -MSE_t \times r\_tune, \quad (15)$$

where MSE_trepresents a function inversely related to the MSE, effectively penalizing higher error values, and $r_tune$is a constant parameter that can be fine-tuned during simulations to facilitate faster convergence of the learning algorithm. This formulation ensures that the reward reflects the system's performance, driving the agent toward minimizing the MSE while achieving the desired trade-offs between speed and accuracy.

6. TD3 ALGORITHM

In this study, we explore the utilization of the TD3 algorithm, a model-free and policy-based DRL approach, to handle the complexities and dynamic nature of the environment. Our proposed framework leverages the TD3 algorithm to efficiently manage continuous state and action spaces, addressing challenges inherent in such systems.

The proposed TD3-based solution incorporates six distinct neural networks, each playing a specific role in the decision-making process. These networks work collaboratively to optimize performance and ensure stability in training. The roles and functionalities of these networks are elaborated as follows:

Actor Network: The actor network, often referred to as the policy network, is a key component responsible for generating actions based on the current state of the environment. It is parameterized $\pi_\phi$and serves as the decision-making entity within the TD3 framework. The actor network maps a given state $S_t$to a corresponding action a_t,enabling the agent to interact with the environment effectively. This process can be mathematically expressed as:

$$a_t = \pi_\phi(S_t) + \zeta, \quad (16)$$

where $\pi_\varphi$represents the policy function parameterized by$\varphi$and ζdenotes a random noise process introduced to encourage exploration of the action space during training. This exploration ensures that the agent does not converge prematurely to suboptimal policies and thoroughly explores the environment.

Two Critic Networks: The TD3 algorithm incorporates two critic networks, often referred to as Q-networks, to evaluate the quality of actions taken by the agent. These networks, parameterized by ϑ_1 and ϑ_2,estimate the Q-value for a given action $a_t$and state $S_t$providing a measure of expected future rewards. The Q-value predictions from the critic networks can be expressed as: $Q_{\vartheta_1}(S_t, a_t; \vartheta_1)$ and $Q_{\vartheta_2}(S_t, a_t; \vartheta_2)$, where $Q_{\vartheta_1}(S_t, a_t; \vartheta_1)$ and $Q_{\vartheta_1}(S_t, a_t; \vartheta_2)$ are the outputs of the two critic networks. Using two critics helps mitigate the overestimation bias commonly found in Q-learning

algorithms, as the TD3 framework selects the minimum of the two Q-values during training to compute the target. This design enhances the stability and reliability of the learning process.

Target Actor Network: The Target Actor Network serves as a prior version of the actor network, distinguished byϕ' as: s. It produces an output with added noise $\zeta$to stabilize the value estimation. This output is clipped to ensure it remains within a defined target range. The network's parameters are periodically refreshed using a soft update strategy, governed by a coefficient, as described below:

$$\phi' \leftarrow \tau\phi + (1-\tau)\phi'. \tag{17}$$

Two Target Critic Networks: These are the earlier iterations of the critic networks, parameterized by $\vartheta_1', \vartheta_2'$. they compute the Q-value $Q_{\vartheta'_i}(S_{t+1}, a_{t+1}; \vartheta_i')$. The parameters are gradually updated over time using a soft update process defined as:

$$\begin{aligned} \vartheta_1' &\leftarrow \tau\vartheta_1 + (1-\tau)\vartheta_1'. \\ \vartheta_2' &\leftarrow \tau\vartheta_2 + (1-\tau)\vartheta_2'. \end{aligned} \tag{18}$$

The actor network is optimized to maximize the objective function through a policy gradient approach, which adjusts the actor's parameters using the following update rule:

$$\nabla_\phi J(\phi) = E[\nabla_{a_t} Q_\vartheta(S_t, a_t)|_{a_t=\pi(S_t)} \nabla_\phi \pi(S_t)]. \tag{19}$$

Simultaneously, the critic networks are trained to minimize the loss function relative to the target value. This process is expressed as:

$$\begin{aligned} L(\vartheta_1) &= E((Y - Q_{\vartheta_1}(S_t, a_t))^2), \\ L(\vartheta_2) &= E((Y - Q_{\vartheta_2}(S_t, a_t))^2). \end{aligned} \tag{20}$$

The target function is defined as:

$$\begin{aligned} &Y = reward + \\ &\iota(min(Q_{\vartheta_1}(S_{t+1}, \pi'(S_{t+1})), Q_{\vartheta 2}(S_{t+1}, \pi'(S_{t+1}))) + \zeta)). \end{aligned} \tag{21}$$

Here, ι denotes the discount factor. Choosing the smaller Q-value from the critics mitigates Q-value overestimation, while adding the noise term to the target policy helps reduce overfitting.

7. Simulation Results

In this section, we present numerical results to demonstrate how integrating MA arrays with the TD3 algorithm can significantly enhance the performance of OTA-FL. The simulation setup assumes that the distances between clients and $l[0]$, uniformly distributed within the range of [20, 100] meters, while the AoAs are uniformly distributed over $[-\pi/2, \pi/2]$ radians. The parameters for the MA array are set as $D_0 = 0.5\lambda$and $D = 8\lambda$, where λ is the wavelength.

For the TD3, SAC, A2C algorithm, the configuration includes a learning rate of 0.0005, a replay buffer size of 10^4, a batch size of 64, a soft update rate of 0.001, and a discount factor of 0.9.

To assess the performance, we conduct two comparisons. First, we compare the MA-based system with a FPA approach, where a fixed location vector $d = [\frac{D}{N+1}, \ldots, \frac{ND}{N+1}]^T$is used. This comparison helps us evaluate the effectiveness of the MA array in improving OTA-FL performance over a more traditional method. Second, we compare the proposed TD3 algorithm with two other reinforcement learning algorithms: SAC and A2C. This comparison is aimed at showcasing the advantages of the TD3 algorithm over these alternative approaches in optimizing the system's performance.

The learning performance is measured by calculating the average rewards over 10 episodes. The average reward for episode b is computed as:

$$reward_{avg}(e) = 0.1 \times \sum_{i=b-10}^{b} reward_i \tag{22}$$

where$reward_i$represents the reward obtained in episode i and b denotes the total number of episodes. This method allows for a thorough evaluation of the TD3 algorithm's performance in optimizing OTA-FL, both when compared to fixed strategies and other reinforcement learning methods. In this simulation, the Baseline3 library in Python is utilized to model and analyze the system dynamics under various operational conditions.

Fig. 1 presents the trend of average rewards for three DRL algorithms, showing a steady increase followed by eventual convergence after 200 training episodes. The SAC algorithm follows a similar convergence pattern to TD3 but achieves lower average rewards when trained with the same learning rate, highlighting the superior performance of TD3. On the other hand, the A2C algorithm converges at a later stage and exhibits consistently lower average rewards throughout the training, suggesting that its performance is inferior compared to both TD3 and SAC. This comparison underscores the effectiveness of the TD3 algorithm in optimizing the learning process.

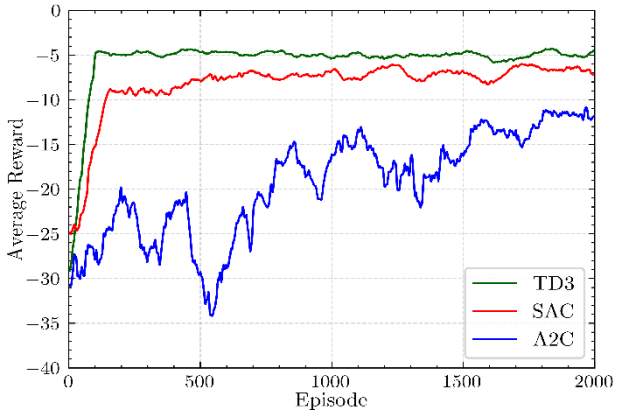


Fig. 1. Comparison of different DRL algorithms versus episodes

Fig. 2 illustrates the relationship between the number of FL clients and the average reward achieved using the TD3 and SAC algorithms in both FPA and MA scenarios. The results show that, for the same number of clients, the proposed TD3 algorithm consistently outperforms SAC across both FPA and MA setups. Additionally, the data indicates that the MA configuration consistently yields better performance compared to the FPA setup, highlighting the advantages of using MA in optimizing the learning process in federated learning environments.

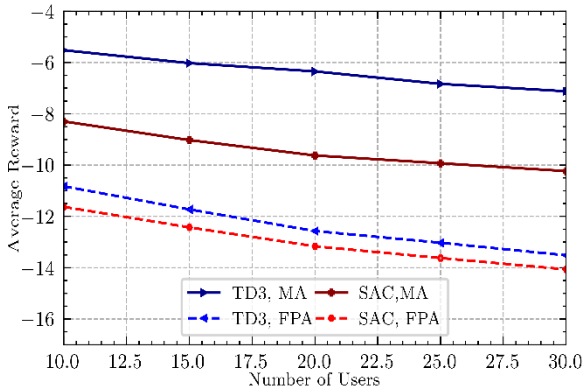


Fig. 2. Comparison of algorithms with different numbers of FL clients

Fig. 3 demonstrates how the effectiveness of the algorithm changes with varying values of K for both SAC and TD3, under the FPA and MA frameworks. In both algorithms, the MA framework consistently shows superior performance compared to the FPA setup. Additionally, TD3 outperforms SAC across both frameworks. As the number of elements K increases, performance initially improves for both algorithms. However, this improvement begins to level off and even diminish as K continues to grow, which is likely due to the degradation in channel quality.

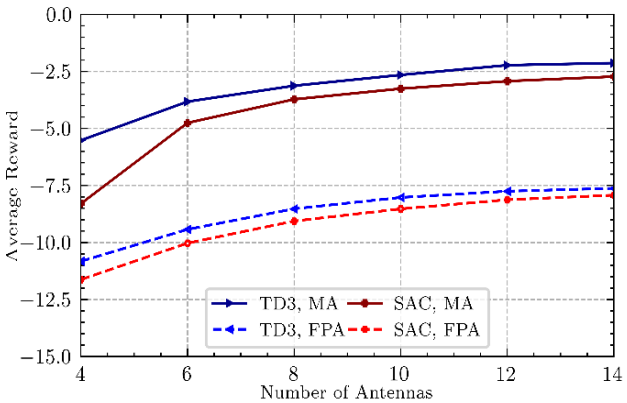


Fig. 3. Comparison of algorithms for different numbers of antennas

For the same values of k, the proposed TD3 algorithm consistently achieves better results than SAC. The diminishing performance enhancement for large k values can be attributed to the limited physical space available when the number of antennas increases while keeping the antenna array length

constant. As the number of antennas grows, the Spacing between them decreases, reducing the diversity and spatial resolution of the system. This limitation leads to a smaller overall performance gain, making the performance difference between the FPA and MA frameworks less pronounced as k becomes large.

To showcase the efficiency of our proposed approach for tackling FL tasks, we trained image classification models on the MNIST dataset. A fully participatory FL framework with five FL clients (N=5), representing UEs, was used. The MNIST dataset was divided into training (90%) and testing (10%) subsets. The training subset was then split into five non-overlapping shards, which were distributed among the clients without replacement. Each client implemented a local model based on a feedforward neural network. The architecture included an input layer with 200 neurons and a ReLU activation function, followed by a hidden layer also containing 200 neurons and a ReLU activation function. The output layer had neurons equal to the number of classes and used a softmax activation function for classification. An ideal FL scenario was considered, without communication noise or interference temperature constraints, to serve as a benchmark.

Error! Reference source not found. illustrates that the TD3 method surpasses SAC in both training loss and test accuracy. As the communication rounds progress, the performance gap between the TD3 method and SAC increases. Additionally, in both algorithms, MA consistently performs better than FPA. This underscores the importance of optimizing the RIS configuration, where phase shift optimization plays a pivotal role in improving overall performance.

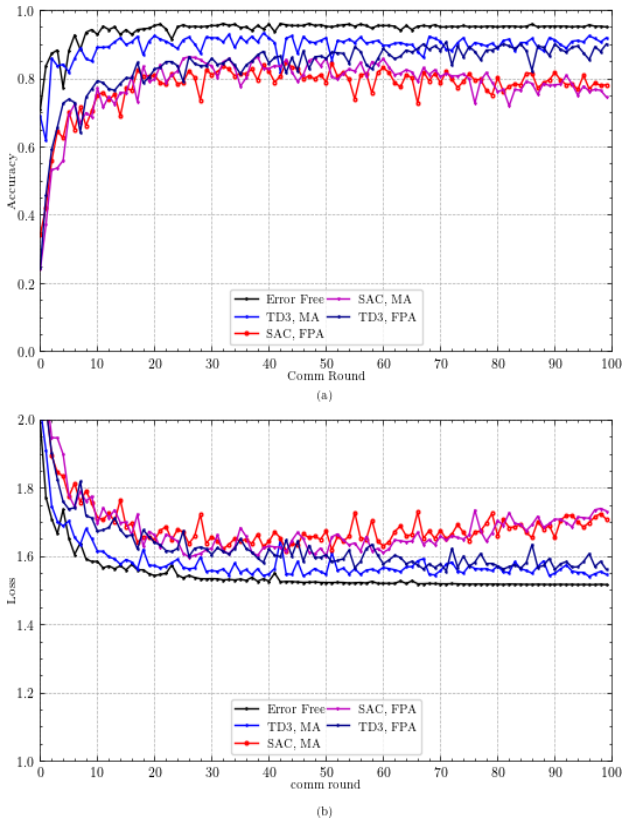


Fig. 4. Comparison of DRL Algorithms on MNIST

8. CONCLUSION

This paper introduced an innovative OTA-FL framework that incorporates MAs at the UAV and UEs as FL clients to significantly enhance system learning efficiency. A non-convex optimization problem was formulated with the goal of minimizing MSE by jointly optimizing antenna placement and beamforming vectors. To account for dynamic environments, this optimization problem was reformulated as a MDP and solved using the TD3 algorithm. The simulation results demonstrated that TD3 outperforms traditional stationary antenna configurations, illustrating the superiority of the proposed MA-based approach. Comparisons between MA arrays and FPA, as well as with alternative algorithms like SAC and A2C, consistently showed that TD3 outperforms FPA, especially as the number of antennas and clients increases. This leads to higher average rewards and significantly better system performance, confirming the effectiveness of the proposed method in enhancing OTA-FL performance.

9. APPENDIX

In the t-th communication round, the global model update is expressed in the following form:

$$\hat{v}_{t+1} = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{\sqrt{\eta}} W^H p_n g_n[d]\, v_n + \frac{w^H z}{N\sqrt{\eta}}, \tag{23}$$

In order to minimize MSE, by applying channel inverse power allocation [22] [4], the power for each client is now determined as follows:

$$p_n = \frac{\sqrt{\eta}}{W^H g_n[d]}, \quad \forall n \in [1,\ldots,N]. \tag{24}$$

By substituting equation (24) into equation (23) we obtain:

$$\begin{aligned}\hat{v}_{t+1} &= \frac{1}{N}\sum_{n=1}^{N} v_{n,t} + \frac{w^H z}{N\sqrt{\eta}},\\ &= \frac{1}{N}\sum_{n=1}^{N}(v_t - \gamma\nabla G(v_t, S_n)) + \frac{w^H z}{N\sqrt{\eta}},\\ &= v_t - \frac{\gamma}{N}\sum_{n=1}^{N}(\nabla G(v_t, S_n)) + \frac{w^H z}{N\sqrt{\eta}},\\ &= v_t - \gamma(\nabla G(v_t)) - \varepsilon_t).\end{aligned} \tag{25}$$

Here, $\nabla G(v_t) = \frac{1}{N}\sum_{n=1}^{N}\nabla G(v_t, S_n)$represents the global gradient, while $\varepsilon_t = \frac{w^H z}{\gamma N\sqrt{\eta}}$refers to the aggregation error.

By incorporating equation (8) and setting $\gamma = \frac{1}{r}$ and taking expectation, we obtain the following expression:

$$\begin{aligned}&E(G(v_{t+1})) - E(G(v_t))\\ &\quad\le \frac{-1}{2r}||\nabla G(v_t)||^2 + \frac{1}{2r}E(|\varepsilon_t|^2),\end{aligned} \tag{26}$$

Now, with some straightforward mathematical simplifications, we obtain:

$$\begin{aligned}&E(G(v_{t+1})) \le E(G(v_t)) -\\ &\frac{1}{2r}||\nabla G(v_t)||^2 + \frac{q|m|^2}{2N^2\eta}E(|z|^2),\end{aligned} \tag{27}$$

By applying the upper bound of the above equation using equation (10) and equation (9), we derive:

$$\begin{aligned}&E(G(v_{t+1})) \le E(G(v_t))\\ &-\frac{\mu}{r}(E(G(v_t)) - E(G^*(v_t))) + \frac{q|m|^2}{2N^2\eta}\sigma^2,\end{aligned} \tag{28}$$

Based on the maximum power constraint for each client and equation (24), ηcan be calculated as follows:

$$\begin{aligned}&\frac{1}{q}\frac{\eta}{|W^H g_n[d]||^2}|E[|v_n|^2]||^2 \le P_{max}\\ &\eta \le \frac{P_{max}}{\Lambda\frac{|W^H g_n[d]|^2}{|w|^2}}\\ &\eta = \min_n \frac{P_{max}}{\Lambda\frac{|W^H g_n[d]|^2}{|w|^2}}\end{aligned} \tag{29}$$

By applying the minimum value of $\eta$and performing some simple mathematical simplifications, we obtain:

$$E(G(v_{t+1})) - E(G(v^*{}_t)) \tag{30}$$

$$+\frac{r\sigma^2\Lambda}{2N^2P_{max}\max_n \frac{|w|^2}{|W^H g_n[d]|^2}}$$

where the second part of the above equation corresponds to the MSE in this scenario:

$$\begin{aligned}
&MSE_t = E[|\hat{v_{t+1}} - v_{t+1}|^2] \\
&= \frac{1}{N^2}\sum_{n=1}^{N} |1 - \frac{1}{\sqrt{\eta_t}} {w_t}^H p_{n,t} g_{n,t}[d]\,|^2 E[|v_{n,t}|^2] \\
&\quad + \frac{q|w|^2\sigma^2}{N^2\eta_t}, \\
&= \frac{q|w|^2\sigma^2}{N^2\eta}, \\
&= \frac{r\sigma^2\Lambda}{2N^2P_{max}\max_{n\in[1,\dots N]} \frac{|w_t|^2}{|{w_t}^H g_{n,t}[d]|^2}}
\end{aligned} \tag{31}$$

Thus, the equation (30) can be rewritten as follows:

$$\begin{aligned}
&E(G(v_{t+1})) - E(G^*(v_t)) \\
&\le (1-\frac{\mu}{r})(E(G(v_t)) - E(G^*(v_t))) \\
&\quad + MSE_t.
\end{aligned} \tag{32}$$

The optimality gap quantifies the discrepancy between the global model's state at the t -th communication round and its initial state, relative to the optimal model. To compute this, we systematically and iteratively apply recursive operations based on the structure outlined in (32). Additionally, by integrating the detailed definition of the MSE provided in (31), we obtain a comprehensive and precise expression for the cumulative optimality gap, which can be represented as follows:

$$\begin{aligned}
&E(G(v_{t+1})) - E(G(v^*)) \\
&\le (1-\frac{\mu}{r})(E(G(v_t)) - E(G(v^*))) \\
&\quad + MSE_t, \\
&\le (1-\frac{\mu}{r})(((1-\frac{\mu}{r}) \\
&\qquad (E(G(v_{t-1}) - E(G(v^*)) \\
&+MSE_{t-1}) - E(G(v^*))) + MSE_t, \\
&\le (1-\frac{\mu}{r})(((1-\frac{\mu}{r})(((1-\frac{\mu}{r}) \\
&(E(G(v_{t-2})) - E(G(v^*))) + MSE_{t-2}) \\
&-E(G(v^*)) + MSE_{t-1}) - E(G(v^*))) \\
&\quad + MSE_t, \\
&\le\dots \\
&\le (1-\frac{\mu}{l})^T (E[G(v_1)] - E[G(v^*)]) \\
&+ \sum_{t=1}^{T} (1-\frac{\mu}{r})^{T-t} MSE_t.
\end{aligned} \tag{33}$$

This concludes the proof of Theorem 1.

10. REFERENCES

[1] M. Ahmadzadeh, S. Pakravan, G. A. Hodtani, M. Zeng, and J. Y. Chouinard. (2024, Nov.). Deep Reinforcement Learning for Robust RIS-Aided Over-the-Air Federated Learning in Cognitive Radio. Presented at 2024 IEEE Middle East Conference on Communications and Networking (MECOM). [Online]. Available: https://doi.org/10.1109/MECOM61498.2024.10881858

[2] M. Pourghasemian, H. Gacanin, and E. Perenda. (2023, Dec.). Cooperative Partial Task-Offloading for Heterogeneous Industrial Robotic MEC System Using Spectral and Energy-Efficient Federated Learning. Presented at IEEE Global Communications Conference (GLOBECOM 2023). [Online]. Available: https://doi.org/10.1109/GLOBECOM54140.2023.10436994

[3] S. Pakravan, M. Ahmadzadeh, M. Zeng, G. A. Hodtani, J. Y. Chouinard, and L. A. Rusch. (2024, Sep.). Robust Resource Allocation for Over-the-Air Computation Networks with Fluid Antenna Array. Presented at IEEE Globecom Workshops (GC Wkshps). [Online]. Available: https://doi.org/10.48550/arXiv.2504.16221

[4] M. Ahmadzadeh, S. Pakravan, G. A. Hodtani, M. Zeng, J. Y. Chouinard, and L. A. Rusch. (2024, Jul.). Enhancement of Over-the-Air Federated Learning by Using AI-based Fluid Antenna System. Presented at arXiv. [Online]. Available: https://doi.org/10.48550/arXiv.2407.03481

[5] Y. Wang, Z. Su, N. Zhang, and A. Benslimane. (2020, Aug.). Learning in the Air: Secure Federated Learning for UAV-Assisted Crowdsensing. *IEEE Transactions on Network Science and Engineering*. [Online]. *8(2)*, pp. 1055–1069. Available: https://doi.org/10.1109/TNSE.2020.3014385

[6] M. Bakhshi, S. H. Ayatollahi, and M. Akbari. (2024). Enhancing Long-Range Radar (LRR) Automotive Applications: Utilizing Metasurface Structures to Improve the Performance of K-band Longitudinal Slot Array Antennas. *AEU - International Journal of Electronics and Communications*. [Online]. *176*, pp. 155134. Available: https://doi.org/10.1016/j.aeue.2024.155134

[7] S. Pakravan, J. Y. Chouinard, X. Li, M. Zeng, W. Hao, Q. V. Pham, and O. A. Dobre. (2023, Jul.). Physical Layer Security for NOMA Systems: Requirements, Issues, and Recommendations. *IEEE Internet of Things Journal*. [Online]. *10(24)*, pp. 21721–21737. Available: https://doi.org/10.1109/JIOT.2023.3296319

[8] M. Bakhshi, S. H. Ayatollahi, and M. Akbari. (2024). Enhanced 2-port MIMO Antenna with Composite Two-Step Metasurface for 77 GHz Vehicle-to-Everything Applications. *AEU - International Journal of Electronics and Communications*. [Online]. *184*, pp. 154982. Available: https://doi.org/10.1016/j.aeue.2024.155404

[9] S. Pakravan, J. Y. Chouinard, M. Zeng, X. Li, W. Hao, and O. A. Dobre. (2023, Dec.). Physical-Layer

Security of RIS-Assisted Networks Over Correlated Fisher-Snedecor F Fading Channels. *IEEE Internet Things J*ournal. [Online]. *11(9)*, pp. 15152–15165. Available: https://doi.org/10.1109/JIOT.2023.3343936

[10] S. Park and H. Seo. (2025, Mar.). Federated Learning Meets Fluid Antenna: Towards Robust and Scalable Edge Intelligence. Presented at arXiv. [Online]. Available: https://doi.org/10.48550/arXiv.2503.03054

[11] L.-H. Shen and Y.-H. Chiu. (2025, Apr.). RIS-Aided Fluid Antenna Array-Mounted AAV Networks. *IEEE Wireless Communications Letters.* [Online]. *14(4)*, pp. 1049–1053. Available: https://doi.org/10.1109/LWC.2025.3531049

[12] M. Pourghasemian, M. R. Abedi, S. S. Hosseini, N. Mokari, M. R. Javan, and E. A. Jorswieck. (2022). AI-Based Mobility-Aware Energy Efficient Resource Allocation and Trajectory Design for NFV Enabled Aerial Networks. *IEEE Transactions on Green Communications and Networking*. [Online]. *7(1)*, pp. 281–297. Available: https://doi.org/10.1109/TGCN.2022.3186911

[13] M. Pourghasemian, H. Gacanin, and E. Perenda. (2023, Dec.). Cooperative Partial Task-Offloading for Heterogeneous Industrial Robotic MEC System Using Spectral and Energy-Efficient Federated Learning. Presented at GLOBECOM 2023 – IEEE Global Communications Conference. [Online]. Available: https://doi.org/10.1109/GLOBECOM54140.2023.10436994

[14] S. Sheikhzadeh, M. Pourghasemian, M. R. Javan, N. Mokari, and E. A. Jorswieck. (2021, Dec.). AI-Based Secure NOMA and Cognitive Radio-Enabled Green Communications: Channel State Information and Battery Value Uncertainties. *IEEE Transactions on Green Communications and Networking*. [Online]. *6(2)*, pp. 1037–1054. Available: https://doi.org/10.1109/TGCN.2021.3135479

[15] J. Yao and N. Ansari. (2021, Dec.). Secure Federated Learning by Power Control for Internet of Drones. *IEEE Transactions on Cognitive Communications and Networking*. [Online]. *7(4)*, pp. 1021–1031. Available: https://doi.org/10.1109/TCCN.2021.3076167

[16] J. Yao and N. Ansari. (2021, Dec.). Secure Federated Learning by Power Control for Internet of Drones. *IEEE Transactions on Cognitive Communications and Networking*. [Online]. *7(4)*, pp. 1021–1031. Available: https://doi.org/10.1109/TCCN.2021.3076167

[17] Q. V. Pham, M. Le, T. Huynh-The, Z. Han, and W. J. Hwang. (2022, May). Energy-Efficient Federated Learning Over UAV-Enabled Wireless Powered Communications. *IEEE Transactions on Vehicular Technology*. [Online]. *71(5)*, pp. 4977–4990. Available: https://doi.org/10.1109/TVT.2022.3150004

[18] Y. Yu, X. Liu, Z. Liu, and T. S. Durrani. (2023, Oct.). Joint Trajectory and Resource Optimization for RIS Assisted UAV Cognitive Radio. *IEEE Transactions on Vehicular Technology*. [Online]. *72(10)*, pp. 13643–13648. Available: https://doi.org/10.1109/TVT.2023.3270313

[19] R. Liu, K. Guo, K. An, F. Zhou, Y. Wu, Y. Huang, and G. Zheng. (2023). Resource Allocation for NOMA-Enabled Cognitive Satellite–UAV–Terrestrial Networks with Imperfect CSI. *IEEE Transactions on Cognitive Communications and Networking*. [Online]. *9(4)*, pp. 963–976. Available: https://doi.org/10.1109/TCCN.2023.3261311

[20] M. Ahmadzadeh, S. Pakravan, and G. A. Hodtani. (2024, Dec.). Movable Antenna Design for UAV-Aided Federated Learning via Deep Reinforcement Learning. Presented at 2024 15th International Conference on Information and Knowledge Technology (IKT). [Online]. Available: https://doi.org/10.1109/IKT65497.2024.10892658

[21] M. Ahmadzadeh, S. Pakravan, and G. A. Hodtani. (2024, Dec.). Movable Antenna Design for UAV-Aided Federated Learning via Deep Reinforcement Learning. In Proceedings of the 15th International Conference on Information and Knowledge Technology (IKT), pp. 91–95.

[22] X. Cao, G. Zhu, J. Xu, and S. Cui. (2022, May). Transmission Power Control for Over-the-Air Federated Averaging at Network Edge. *IEEE Journal on Selected Areas in Communications*. [Online]. *40(5)*, pp. 1571–1586. Available: HTTPS://DOI.ORG/10.1109/JSAC.2022.3143217

Journal of Computer and Knowledge Engineering, Vol. 8, No. 2, 2025. (10-24)

Ferdowsi University of Mashhad

Journal of Computer and Knowledge Engineering

https://cke.um.ac.ir

Information and Communication Technology Association of Iran

Efficient Implementation of DVI Protocol on FPGA*

Research Article

Sara Ershadi-Nasab[1], Danial Bayati[2], Saeed Yazdani[3]

10.22067/cke.2025.91345.1142

Abstract *This paper presents a general-purpose hardware implementation of the digital visual interface (DVI) protocol on the Xilinx Virtex-6 ML605 FPGA platform for real-time display of digital processing results. The design enables direct output of processed data from the FPGA to an external monitor without relying on external processors or software-based rendering tools. It addresses key challenges in timing synchronization, pixel formatting, and interfacing with the onboard Chrontel CH7301C encoder to support resolutions up to 1920×1080 at 60 Hz. A lightweight processing pipeline is developed in Verilog to convert multidimensional outputs into a sequential stream of pixel data conforming to the DVI protocol. As a case study, a lightweight convolutional neural network trained on the CIFAR-10 dataset is implemented on the FPGA, and its classification probabilities are displayed as a probability map on an LCD. Experimental results confirm low resource utilization and real-time performance, validating the system's applicability in embedded applications such as machine learning inference, image processing, and real-time monitoring. This work demonstrates the feasibility of FPGA- based platforms for efficiently displaying digital video output in intelligent edge systems.*

Key Words Digital visual interface, field programmable gate array, image processing, real-time, video processing, Xilinx Virtex-6.

1. INTRODUCTION

THE integration of neural networks with real-time image processing on FPGA platforms is vital for applications that require low latency and efficient computation. While neural networks are effective at interpreting visual data, displaying their classification results in real time directly from an FPGA presents practical challenges. In software environments such as Python or MATLAB, classification results can be easily shown using high-level functions like plot(). However, FPGA-based systems lack such abstractions; displaying outputs requires manual control over video signal generation and transmission to an external monitor. This makes even basic output display—such as showing the predicted class on an LCD screen—a nontrivial hardware task.

The DVI is a widely accepted standard for transmitting digital video signals, enabling seamless connections between computers and display devices [1], [2]. It requires careful synchronization of pixel data, clock signals, and timing signals to ensure the correct display of visual outputs. Misalignment of these signals can result in distorted or invisible outputs. Therefore, a precise and accurate implementation of the DVI protocol is crucial for verifying the functionality of FPGA- based neural network systems. As the demand for high- quality digital video transmission increases, bridging the gap between legacy interfaces—such as the video graphics array (VGA) and modern digital standards has become essential. Field-programmable gate arrays (FPGAs) provide a flexible and customizable platform for implementing such interface conversions.

In modern image processing and deep learning applications, it is often necessary to display the output of computations on a monitor [3], [4]. This requirement becomes particularly important when processing is carried out on FPGA boards, as the FPGA functions as a specialized integrated circuit (IC) that performs intensive computational tasks. The results must be presented clearly and efficiently to ensure that the system operates as intended. In recent years, implementing deep learning algorithms on FPGA boards has gained significant attention, as these platforms offer high performance, low power consumption, and customizability compared to traditional processors [5], [6], [7]. Many embedded and

* Manuscript received 2024 December 29, Revised 2025 January 24, Accepted 2025 May 31.

[1] Corresponding author, Assistant Professor, Computer engineering department, Ferdowsi University of Mashhad, Mashhad, Iran. **Email:**ershadinasab@um.ac.ir.

[2] M.Sc. Student. Computer engineering department, Ferdowsi University of Mashhad, Mashhad, Iran.

[3] M.Sc. student. Computer engineering department, Ferdowsi University of Mashhad, Mashhad, Iran.

edge applications now utilize specialized ICs, rather than general-purpose processors, to per- form neural network inference. These specialized ICs optimize power, memory usage, and transistor count, making them well- suited for real-time processing. However, to verify the accuracy and effectiveness of neural networks, the processed outputs must be displayed to the user in a comprehensible manner, necessitating the implementation of a video display protocol.

A key limitation in FPGA-based video systems arises from the interface between multidimensional data structures used in AI models and the strictly sequential data format required by video output hardware. While the Chrontel (CH7301C) [8] DVI encoder integrated on the ML605 board handles the encoding of pixel data into the transition minimized differential signaling (TMDS) format for robust transmission over DVI cables, it requires a continuous, precisely timed stream of pixel values along with horizontal and vertical synchronization signals. In contrast, neural networks and image processing algorithms typically operate on two- or three-dimensional data arrays—such as feature maps or RGB images—rather than linear pixel sequences. This structural mismatch demands custom logic to convert high-level, array-based outputs into tightly timed, flattened pixel streams that comply with the DVI protocol. The lack of built-in hardware for this conversion makes it necessary to implement a fully synchronized pipeline capable of feeding the Chrontel encoder accurately and in real time. Addressing this challenge is essential for achieving smooth and coherent display of neural network results in embedded artificial intelligence (AI) applications. This paper presents an efficient hardware-software co-design approach for real-time display of neural network classification results using the DVI protocol on an FPGA. A lightweight neural network, trained on the CIFAR-10 dataset, is implemented within the FPGA, with its weights manually embedded in Verilog. The network processes input images and produces classification probabilities, from which the seven most probable classes are selected for display. These results are formatted as RGB data, which is then converted into a DVI-compatible signal for output to an external monitor. To achieve this, we generate synchronization signals, overlay the classification results on the display, and transmit the output to the onboard Chrontel DVI encoder of the Xilinx ML605 board. This enables real- time rendering of the classification output directly on an LCD screen, alongside the processed input image. By eliminating the need for external CPUs or GPUs, this approach enhances resource efficiency and provides a scalable solution for embedded AI applications. The proposed system demonstrates how FPGA-based deep learning architectures can be directly integrated with standard video output protocols, making them well-suited for edge computing and real-time classification tasks. This paper discusses the process of implementing the DVI protocol on the ML605 FPGA, including the challenges of signal synchronization, the integration of neural network output, and the successful testing of the system for real-time image processing applications. By utilizing the power of the Xilinx Virtex-6 architecture, we aim to showcase the potential of FPGA platforms in the efficient and accurate display of neural network and image processing outputs.

One of the significant challenges in the integration of neural network outputs and real-time image processing in FPGA systems is the precise synchronization of timing signals for video display, especially in legacy-to-modern interface conversions. Existing solutions often focus on individual aspects, such as processing efficiency or display fidelity, without addressing the combined demand for real-time performance and high- quality visualization. Our work introduces a novel approach by integrating DVI protocol on the ML605 FPGA to achieve seamless and accurate displaying of neural network outputs. The system leverages the capabilities of the Xilinx Virtex-6 architecture, ensuring synchronization of pixel, clock, and timing signals with resolutions up to 1920×1080 at 60 Hz. This dual capability of high-resolution rendering and real-time processing represents a significant step forward in the design of embedded systems, particularly for edge AI applications like medical imaging and smart monitoring. In this work, the ML605 FPGA development board, built on the Xilinx Virtex-6 architecture, is employed to prototype and implement complex digital systems. The design is developed using the Xilinx ISE 14.7 toolchain, which fully supports hardware synthesis and configuration for this platform. With its high-capacity FPGA, ample memory, and versatile I/O interfaces, the ML605 is well-suited for system-on-chip (SoC) development and the implementation of advanced video interfaces.

The primary contribution of this work is the real-time implementation of the DVI protocol on the Xilinx Virtex-6 ML605 FPGA platform. This implementation enables direct displaying of neural network outputs and image processing results by addressing key challenges in signal synchronization and high-resolution rendering. Key innovative aspects of the implementation include:

1) **Real-time DVI protocol implementation on FPGA:** This work presents the complete implementation of the DVI protocol on the Xilinx ML605 FPGA board, enabling direct display of neural network outputs on external monitors without requiring a host CPU or GPU.
2) **Direct integration of neural inference with display pipeline:** A novel hardware-software co-design is developed that links a lightweight convolutional neural network (CNN), manually implemented in Verilog, with a real-time pixel stream generator. The system converts multidimensional classification outputs into sequential RGB pixel values for video rendering.
3) **Custom hardware pipeline for TMDS-compatible output:** The system addresses the structural mismatch between array-based AI outputs and the linearly timed pixel streams required by the DVI encoder. A synchronized pipeline is implemented to format and transmit the CNN output via the on-board Chrontel CH7301C encoder using the TMDS standard.
4) **Legacy-to-modern interface conversion:** The design enables legacy VGA-style embedded systems to connect with modern digital displays via the DVI protocol, effectively bridging analog and digital video standards using programmable logic.

5) **Efficient FPGA resource utilization:** The architecture demonstrates minimal consumption of FPGA resources (less than 1% of slice registers and LUTs), allowing significant headroom for additional logic, such as more complex networks or preprocessing units.
6) **Hardware-based rendering of neural outputs:** Classification probabilities are displayed as visual indicators—such as variable-width bars—directly on the LCD without software-based rendering. This hardware-centric approach facilitates real-time feedback in embedded systems.
7) **Scalability for edge AI applications:** The system is well-suited for embedded AI scenarios, such as medical diagnostics, smart surveillance, and industrial monitoring, where low latency, power efficiency, and real-time result visualization are critical.

This novel implementation not only confirms the feasibility of video protocol integration on FPGA platforms but also provides a scalable framework for real-time display in advanced embedded systems.

The structure of this paper is organized as follows: Section II introduces relevant studies addressing the topic. Section III presents a detailed comparison of commonly used display interfaces, including VGA, DVI, and high-definition multimedia interface (HDMI), highlighting their features, limitations, and use cases. The proposed method architecture is outlined in Section IV. The experimental results are provided in Section V. Finally, the conclusion is presented in Section VI.

2. RELATED WORK

Integrating neural networks and real-time image processing outputs on FPGA platforms has garnered significant attention in recent years [9], [5]. Various studies have explored FPGA- based solutions for implementing video display protocols, focusing on DVI and related technologies [10], [9]. These efforts highlight the versatility of FPGAs in bridging legacy and modern systems while ensuring high-performance real- time processing. Recent advancements in AI have driven significant progress in intelligent flexible sensing systems capable of highly efficient data acquisition, analysis, and perception. These innovations enable more sophisticated communication between neural processing units and external sensors, improving real-time monitoring and display capabilities for applications such as flexible sensory systems, humanoid robotics, and human activity monitoring [4]. Similarly, the development of systems-on-chip such as TinyVers, which incorporates state- retentive design for machine learning (ML) inference at the extreme edge, demonstrates the significance of energy-efficient and versatile hardware platforms in supporting real-time AI applications [5]. Recent advancements have showcased the implementation of FPGA-based real-time image processing systems, emphasizing the integration of DVI-compatible video interfaces for effective visualization and synchronization [9]. Optimization techniques for deploying CNNs on FPGA plat- forms have further enhanced the efficiency of hardware- software co-design, enabling seamless display of neural net- work output [10]. High-speed video processing and display integration have also been demonstrated, particularly through FPGA-accelerated object detection systems utilizing edge in-formation, achieving real-time potential in applications [11].

Moreover, digital oscillatory neural network frameworks have been implemented on FPGAs for edge AI applications, highlighting the relevance of video signal generation capabilities in DVI-based visualization systems [12]. The development of real-time systems for processing neuronal network activity on FPGA platforms has further established the critical role of DVI in rendering real-time outputs for high-speed visual feedback [13]. Additionally, FPGA implementations of hyperchaotic neural network systems have illustrated the adaptability of these platforms for complex computations and their corresponding outputs [14].

Efforts have also focused on privacy-preserving authentication protocols for IoT devices, leveraging FPGA capabilities with DVI for secure and efficient visualization [3]. Finally, real-time video enhancement algorithms implemented on FP- GAs have underscored their ability to handle computationally intensive tasks while adhering to DVI standards for high- quality visualization [15].

Another important area of exploration has been the evolution of GPU hardware, which offers significant parallel processing capabilities for neural networks and AI workloads. Peddie [6] provides an in-depth review of the GPU environment and its impact on hardware, highlighting advancements in graphics processing technology and its integration into AI and ML workflows.

These insights are particularly relevant as GPUs and FPGAs continue to coexist as complementary technologies in real-time AI processing. These studies collectively demonstrate the flexibility and efficiency of FPGA platforms in integrating neural networks, real-time image processing, and video display protocols such as DVI. They provide a solid foundation for developing FPGA-based systems that deliver high-speed, accurate visual outputs—crucial for applications in AI, edge computing, and embedded systems.

Wang and Luo [16] emphasize the benefits of FPGA accelerators in optimizing custom hardware architectures for real-time applications. Their review highlights the importance of precision reduction techniques in minimizing latency and enhancing performance—approaches that directly support our objective of achieving high-speed and accurate displaying of neural network outputs.

Recent investigations into FPGA-based visualization systems have also explored optimization strategies aimed at reducing latency and power consumption. In particular, hybrid systems that combine FPGAs with processors or GPUs have received attention for their ability to offload specific tasks, such as preprocessing or feature extraction, to dedicated hardware blocks. This co-design approach is instrumental in meeting the strict timing constraints required for real-time video out- puts [17].

Overall, these studies underscore the potential of FPGA platforms for efficient video protocol integration and real-

time visualization, thereby paving the way for advancements in embedded display systems across applications like medical imaging, edge AI, and smart monitoring. Furthermore, the adaptation of FPGAs for AI-driven video analytics has shown remarkable results. Thyagarajan et al. [18] and Park et al. [15] demonstrated the integration of neural network models with smart cameras, achieving real-time performance in applications like sports analytics and video enhancement. These findings align with the growing need for low-latency, high-throughput FPGA solutions for video-based AI applications. Przesmycki and Nowosielski [19] explored the security implications of compromising emanations in VGA and DVI interfaces, providing insights into the design of secure and reliable FPGA-based visualization systems.

Moreover, Bailey [7] elaborated on the fundamentals of embedded image processing systems on FPGAs, detailing the integration of advanced display protocols like DVI and HDMI for multimedia applications. Hoang et al. [20] presented a pulse-coupled neural network (PCNN) framework implemented on FPGAs for real-time object recognition, showcasing DVI compatibility for visual outputs. Similarly, Fang et al. [21] proposed systematic optimization of spiking neural networks (SNNs) on FPGAs, emphasizing their ability to handle cognitive tasks in real-time scenarios. Farabet et al. [22] developed FPGA-based stream processors for convolutional neural networks, enabling real-time vision tasks with standard DVI connections for video display. These systems demonstrate the capacity of FPGA-based platforms to manage complex visual processing pipelines while ensuring low latency. Additionally, Yildiz et al. [23] and Kayaer et al. [24] explored FPGA implementations of cellular neural networks for preprocessing blocks in high-definition video applications. Their systems utilize DVI interfaces to process and visualize outputs in real time. Abernot [25] investigated the use of oscillatory neural networks on FPGA platforms, highlighting their utility in edge AI systems requiring real-time video processing. This study underscores the adaptability of FPGA designs in integrating learning models with real-time video outputs. Yildiz et al. [26] presented the implementation of preprocessing blocks for cellular neural network-based systems on FPGAs, utilizing DVI for real-time output visualization. This research high- lighted the efficiency of FPGA designs for low-latency video applications. Antonik [27] explored FPGA implementations for hardware reservoir computing and real-time machine learning, emphasizing applications in edge AI. Similarly, Ahilan and James [28] focused on the design and implementation of a real-time car theft detection system, which leveraged FPGA processing and DVI visualization to achieve high-speed image analysis. Davutoğlu et al. [29] designed a real-time frame buffer implementation using external memory on FPGAs. Their study demonstrated how FPGAs can efficiently manage frame data while supporting DVI interfaces for video display. Fasih et al. [30] examined FPGA-based systems for video enhancement in advanced driver assistance systems (ADAS), incorporating convolutional neural networks and DVI outputs to improve video clarity.

TABLE 1 Detailed comparison of Vga, Dvi, and hdmi display interfaces

Feature	**VGA**	**DVI**	**HDMI**
Year of Introduction	1987	1999	2003
Signal Type	Analog	Analog & Digital (DVI-A, DVI-D, DVI-I)	Digital
Maximum Resolution	Up to 1080p	1920x1200 (Single-Link), 2560x1600 (Dual-Link)	8K at 60Hz or 4K at 120Hz (HDMI 2.1)
Color Depth	Limited by analog quality	24-bit (Single-Link) or higher for Dual-Link	Up to 48-bit (HDR supported)
Audio Support	No	No	Yes, with multichannel audio support
Cable Length	Up to 15m with quality loss	Up to 5m for digital, longer for analog	Up to 15m for 4K, shorter for 8K
Compatibility	Legacy monitors and projectors	Transitional systems	Modern displays, TVs, and projectors
Connector Type	15-pin D-Sub	Multi-pin (varied)	Compact (Type-A, Mini, Micro)
Video Signal Quality	Prone to interference	Better than VGA; pure digital avoids noise	Excellent; supports HDR and high refresh rates
Multi-Monitor Support	Not supported	Not supported	Supported via splitters
Data Bandwidth	Not standardized	4.95 Gbps (Single-Link), 9.9 Gbps (Dual-Link)	Up to 48 Gbps (HDMI 2.1)
Use Cases	Legacy monitors and projectors	PC monitors and transitional setups	TVs, gaming systems, multimedia devices
Adapter Availability	VGA to HDMI/DVI with converters	DVI to VGA/HDMI with converters	HDMI to VGA/DVI with converters
Cost	Low	Moderate	Higher (for high-speed cables)

3. COMPARISON OF VGA, DVI, AND HDMI IN FPGA-BASED SYSTEMS

Video interfaces play a crucial role in FPGA-based image processing and displaying the neural network output. Among the widely used standards, VGA, DVI, and HDMI offer different trade-offs in terms of signal quality, bandwidth, and implementation complexity. Table 1. summarizes their key differences, focusing on their impact on FPGA implementation. DVI strikes a balance between complexity and quality, making it a suitable choice for FPGA-based real-time visualization of neural network outputs. Unlike VGA, it provides lossless digital transmission, and compared to HDMI, it avoids the additional complexity of audio and high-bandwidth digital content protection (HDCP) encryption, which are unnecessary for many FPGA applications.

4. PROPOSED METHOD

Fig. 1 illustrates the high-level design of the proposed hardware-based image classification pipeline. This system con- sists of six sequential stages that transform raw image data into classified and visualized output, ultimately displayed in

real time on a digital monitor. The process begins with image processing, where the input image—typically in $W \times H \times 3$ RGB format—is resized, normalized, and optionally filtered to enhance its features and ensure consistency for the classification model. Next, the AI processing stage applies a lightweight classification algorithm, which may be based on neural networks or simpler machine learning methods. This stage extracts relevant features and produces a classification output in the form of a probability map or class index.

The third stage, probability map visualization, converts the AI output into a visual format by mapping probabilities or class indicators into color-coded RGB values. This makes the classification interpretable when displayed on a screen. In stage four, the system generates essential synchronization signals such as horizontal sync (HSYNC), vertical sync (VSYNC), and data enable (DE), along with precise pixel timing to prepare the image stream for display output. These signals ensure that the display device receives video data in a valid scanline order.

Once the pixel data and control signals are properly formatted, the fifth stage interfaces with the Chrontel (CH7301C) DVI encoder chip. This stage handles the conversion of parallel RGB data into TMDS (Transition Minimized Differential Signaling) format, which is the standard for DVI transmission. The sixth and final stage handles the actual DVI output, transmitting the TMDS signals to an external monitor where the classified image is rendered in real time, enabling immediate feedback and visualization.

Fig. 2 provides a detailed RTL schematic that corresponds specifically to the first four stages of the pipeline. These stages are fully implemented in Verilog and deployed on an FPGA platform. The image processing logic is handled by the image_processing module, which receives and formats the incoming image data. This is followed by the ai_processing module, which performs classification using MAC operations and feature extraction based on preloaded filters. The output of this stage is passed to the probability_map_visualizing module, which trans- forms the classification results into pixel-level RGB values based on scan positions.

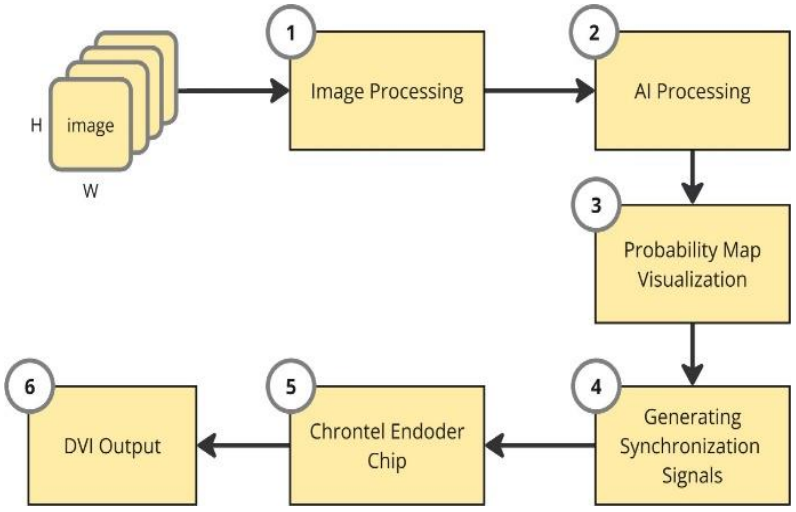


Fig. 1. Overview of the hardware-based image classification pipeline implemented in Verilog. The process is divided into six main stages, from image input to real-time DVI display

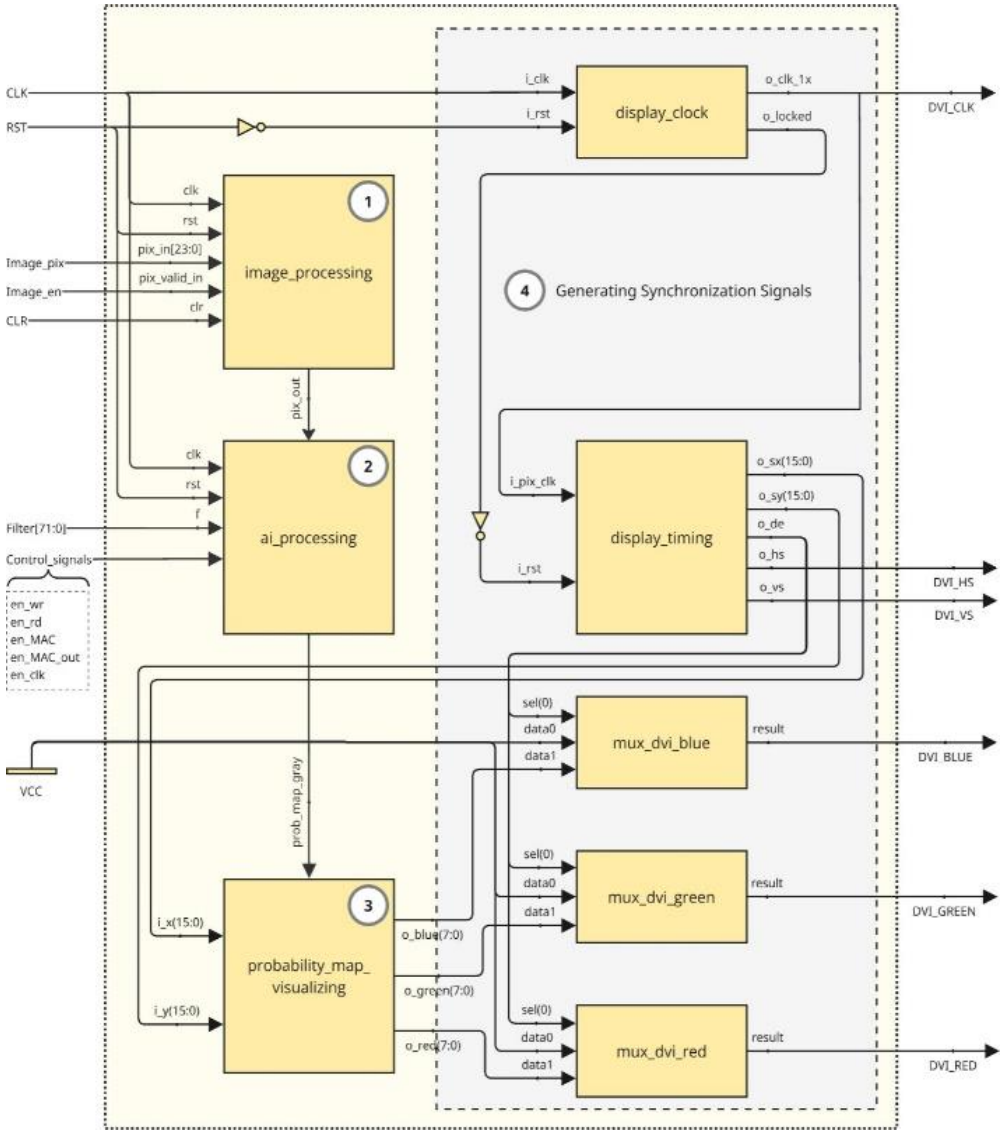


Fig. 2. Register transfer level (RTL) schematic of the Verilog-based image classification system. This diagram focuses on the implementation of Stages 1 through 4 within the FPGA, showing signal flow, control logic, and synchronization modules

To prepare for external video transmission, synchronization signals and timing control are generated by the display_clock and display_timing modules. These provide the pixel clock, scan coordinates, and sync signals required by the downstream encoder. While the RTL diagram concludes at this point, the outputs from Stage 4 are structured specifically for interfacing with the Chrontel encoder in Stage 5, and eventually, real-time rendering on a monitor via Stage 6.

4.1. Image Processing Stage

The Image Processing stage serves as the initial component of our FPGA-based classification pipeline, tasked primarily with preparing raw image data for subsequent analysis by the AI Processing module. This stage plays a fundamental role, ensuring compatibility and quality enhancement of image data, thus directly influencing inference accuracy and computational efficiency.

In the present design, we adopt the widely recognized CIFAR-10 dataset as the primary source of training and evaluation images. CIFAR-10 provides 60,000 images (32×32 pixels, RGB) split into 50,000 training images and 10,000 test images. These images span 10 distinct classes, each containing

6,000 samples. While CIFAR-10 images are natively in color (3 channels).

Initially, input image data arrives in standard RGB format, represented as three separate channels (Red, Green and Blue) with each pixel typically stored at 8-bit color depth. Given the resource constraints and processing requirements of FPGA

hardware, these images undergo several preprocessing steps to enable efficient inference and maintain acceptable accuracy. First, input images are resized to a fixed, uniform resolution compatible with the downstream inference engine

(e.g., 32×32 or 64×64 pixels). This resizing, implemented in Verilog, utilizes hardware-optimized interpolation algorithms

(e.g., bilinear interpolation) to maintain image quality while reducing computational overhead. The choice of a

relatively small, standardized resolution aligns well with the limited memory and computational bandwidth on FPGAs, ensuring predictable timing and efficient parallelization.

Subsequently, normalization of pixel values scales the image data into a suitable numerical range (such as 0–1 in floating- point or Qm.n in fixed-point) to ensure stable arithmetic operations during neural network inference. For this project, an 8-bit RGB input is often transformed into a fixed-point representation (e.g., Q8.8) or scaled floating-point format that fits the FPGA's DSP slices and LUTs. This consistent input magnitude fosters stable training convergence (if on-FPGA training or partial re-training is used) and more accurate inference under resource constraints.

Depending on deployment needs, noise reduction filtering, such as median filtering or Gaussian smoothing, can be added to enhance the signal-to-noise ratio of raw images. In an FPGA context, these filters can be efficiently realized via parallelized convolution modules or simplified averaging techniques. The hardware-level parallelism offered by FPGAs significantly reduces latency for such operations, crucial for real-time applications.

Finally, the processed and normalized image data is buffered in on-chip block RAM or external memory, ready for rapid retrieval during inference. This buffering ensures a smooth pipeline from raw data ingestion to the AI Processing stage, mitigating bandwidth bottlenecks and guaranteeing real-time performance. By streamlining the raw images into a predictable format, the Image Processing stage lays the ground- work for the subsequent hardware-accelerated CNN inference.

4.2. AI Processing Stage

The AI processing stage is the core of our FPGA-based image classification pipeline, where a CNN is implemented directly in Verilog HDL to achieve efficient real-time inference. Leveraging the intrinsic parallel processing capabilities of FPGAs, this design tackles the computationally intensive nature of CNNs while working under the logic, DSP, and memory constraints of devices like the ML605 board.

In our approach, we adopt a six-layer CNN architecture inspired by the work in [31]. The model comprises:

1) Sliding Window Convolution (for feature extraction),
2) ReLU Activation (introducing non-linearity),
3) Max Pooling (down sampling to reduce spatial dimension),
4) Flattening (restructuring 2D features into a 1D vector),
5) Fully Connected (learning global relationships among features),
6) Softmax Activation (producing a probability distribution over the 10 CIFAR-10 classes).

Each layer is coded as a separate Verilog module, allowing straightforward testing and debugging. For instance, the con- volution layer involves efficient hardware-based matrix multi- plication to convolve filters over the input feature maps, while the ReLU module employs a simple conditional operation to clamp negative values to zero. The Max Pooling module further reduces data dimensionality by selecting the maximum value within local neighborhoods of a feature map, improving robustness to minor shifts. Flattening modules then reshape the 2D feature maps into 1D vectors for fully connected processing, and a final Softmax step converts outputs to class probabilities.

Fig. 3 illustrates the internal organization of the inference module. A dedicated memory controller retrieves pretrained weights and biases from off-chip memory (e.g., DDR3 on the FPGA), while the computation engine executes multiply- and-accumulate (MAC) operations in parallel. By instantiating multiple DSP slices for simultaneous MAC operations, the inference pipeline substantially reduces latency compared to software-based implementations.

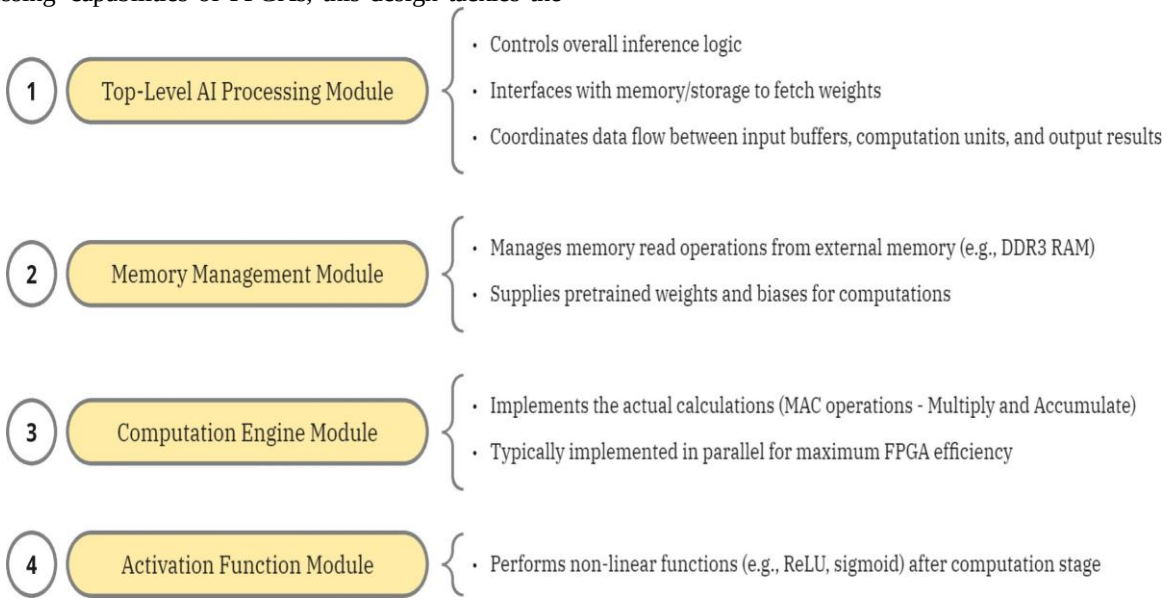


Fig. 3. AI processing stage (inference) module organization.

To achieve consistent performance and accuracy, we pre- trained the CNN offline, using standard frameworks (e.g., Ten- sorFlow or PyTorch) with the CIFAR-10 dataset. During this training phase, high-level floating-point arithmetic was used. Post-training, model parameters were quantized or scaled to fit the fixed-point precision supported by the Verilog modules on the FPGA. This quantization can be as coarse as Q4.12 or Q8.8, depending on resource availability and desired accuracy. An activation function module applies nonlinearities such as ReLU. Compared to sigmoid or tanh, ReLU is both simpler to implement and less prone to saturating at extremes. The AI inference controller, operating in concert with these modules, handles synchronization, data flow, and final classification result generation. Once classification is complete, the output is forwarded to subsequent logic interfaces for digital video output or further processing steps.

By consolidating these hardware modules, we demonstrate a feasible CNN pipeline capable of real-time classification, even on mid-range FPGA platforms. This tightly integrated design exemplifies how FPGAs can address demanding edge inference tasks, combining low-power consumption with competitive throughput for resource-constrained environments. Future enhancements may explore deeper CNN architectures or color- image pipelines once resource usage is further optimized. Nonetheless, the current 6-layer CNN exhibits strong proof- of-concept for FPGA-based deep learning inference on the CIFAR-10 dataset.

4.3. Probability Map Visualization Stage

The probability map visualization stage serves as a critical intermediary within the FPGA-based classification pipeline, connecting AI-generated outputs to the digital video display subsystem. In this stage, the numeric classification results produced by the AI inference engine are systematically converted into clearly distinguishable visual representations suitable for subsequent DVI output. Specifically, classification labels or inference results—initially represented as numeric vectors or encoded class identifiers—are mapped into a predefined color- coding scheme. In our implementation, this involves associating each classified category with a unique color, employing a fixed set of up to ten distinct colors, each corresponding directly to a specific classification result. Such mapping is efficiently realized using lookup tables (LUTs) implemented directly within Verilog code.

The Verilog implementation of the probability map visualization module involves defining a lookup table that associates each of the classification outputs with a preselected RGB color value. This enables immediate, visual differentiation of predicted classes on screen, enhancing interpretability and facilitating rapid decision-making. The hardware module utilizes internal FPGA resources, typically block RAM or distributed LUTs, to perform this quick mapping operation. For instance, an inference result labeled as “Class 1” might correspond to red, while “Class 2” might be displayed as green, and so forth. If the AI inference identifies more than a limited number of classes, a hierarchical encoding strategy can be employed to group classes into broader categories, preserving FPGA resources while maintaining visual clarity.

Additionally, the module handles synchronization tasks necessary for digital video interfaces. It ensures that the converted RGB data stream matches the timing requirements of the DVI protocol, performing frame buffering and pixel synchronization. Precise control of pixel timing, horizontal and vertical synchronization signals, and other required digital video parameters guarantees a stable, high-quality visual output free from artifacts or latency issues.

Moreover, for scenarios involving uncertainty or unidentified classes, an additional category (often represented by a neutral color or grayscale) can be assigned. Implementing

these visual encoding schemes directly through hardware de- scription language allows for seamless, real-time visualization of classification results without latency overhead, which is critical for applications requiring immediate feedback, such as real-time image classification and monitoring systems.

4.4. Generating Synchronization Signals

Stage 4 of the system comprises several essential mod-ules that work in tandem to generate synchronization signals and drive display output. These modules include dis- play_clock, display_timing, and RGB color channel multiplexers (mux_dvi_red, mux_dvi_green, mux_dvi_blue). Together, they handle the timing, pixel positioning, and output format- ting necessary for DVI video transmission, as shown in fig. 2.

The display_clock module is responsible for producing the required video clocks using a mixed-mode clock manager (MMCM). It generates a stable pixel clock (o_clk_1×) for the entire display pipeline and asserts a o_locked signal once clock stabilization is achieved. By manipulating multiplication and division factors, the module ensures that clock frequencies align precisely with resolution-specific requirements, facilitating smooth video playback.

As shown in Table 2, the clock parameters are resolution dependent. The MULT_MASTER parameter sets the base frequency multiplier, while DIV_MASTER, DIV_5×, and DIV_1× divide the result to generate the final pixel clock.

For high-definition formats like 1920×1080, smaller division values (e.g., DIV_1×, × = 5) ensure the required high- frequency clocks are achieved for dense pixel grids.

TABLE 2

Clock setting for different resoulutions

Parameter	**640×480**	**800×600**	**1280×720**	**1920×1080**
MULT_MASTER	31.5	10.0	37.125	37.125
DIV_MASTER	5	1	5	5
DIV_5×	5.0	5.0	2.0	1.0
DIV_1×	25	25	10	5

The display_timing module generates horizontal and vertical sync signals (o_hs, o_vs), display enable (o_de),

and current pixel coordinates (o_sx, o_sy) based on the incoming pixel clock (i_pix_clk). These signals are fundamental for precise raster scanning and timing alignment with modern DVI dis- plays.

Table 3. presents the horizontal and vertical timing parameters. These include the active resolution (H_RES, V_RES) as well as blanking intervals (H_FP, H_SYNC, H_BP, and their vertical counterparts). At higher resolutions like

1920×1080, longer back porch values (e.g., H_BP = 148) allow more time for processing and synchronization. Sync

polarities (H_POL, V_POL) also adapt to modern display requirements—switching to active-high signals for resolutions 800x600 and above.

The output of the display_timing module directly drives the RGB multiplexers: mux_dvi_red, mux_dvi_green, mux_dvi_blue. These modules select between raw image data and AI-generated overlay visuals (received from upstream modules like probability_map_visualizing) using select signals. The chosen color values are then output to the DVI lines: DVI_RED, DVI_GREEN, and DVI_BLUE.

TABLE 3

Display timings for different resoulutions

Parameter	**640×480**	**800×600**	**1280×720**	**1920×1080**
H_RES	640	800	1280	1920
V_RES	480	600	720	1080
H_FP	16	40	110	88
H_SYNC	96	128	40	44
H_BP	48	88	220	148
V_FP	10	1	5	4
V_SYNC	2	4	5	5
V_BP	33	23	20	36
H_POL	0	1	1	1
V_POL	0	1	1	1

4.5. Chrontel Encoder Chip (CH7301C) and DVI Output Stage

The final and crucial step in the FPGA-based image classification pipeline involves the Chrontel (CH7301C) DVI Transmitter chip. This integrated circuit, utilized specifically on the ML605 FPGA development board, is designed for converting digital image data from the FPGA's internal processing units into standardized DVI signals, suitable for high-quality video output.

The Chrontel (CH7301C) is a specialized semiconductor device that accepts parallel digital data (typically in RGB format) from the FPGA and converts it into a serialized digital signal conforming to DVI standards. This chip facilitates the transition from internal FPGA processing outputs into a standard video signal suitable for displays. To achieve this, the (CH7301C) device internally incorporates encoding logic, parallel-to-serial conversion circuits, and synchronization logic. It receives 24-bit parallel RGB data signals along with synchronization signals (horizontal sync, vertical sync, and pixel clock signals) directly from the FPGA output pins. The device then performs parallel-to-serial data conversion, encoding the video data using the TMDS protocol, which is fundamental to the DVI standard.

In practice, once the AI inference results have been converted into color-coded pixel data by the probability map visualization module within the FPGA, these parallel pixel data streams are passed directly into the (CH7301C). This chip organizes the incoming RGB digital signals, applies necessary timing and synchronization adjustments, and serializes the data into TMDS-compliant signals. TMDS encoding ensures minimal electromagnetic interference (EMI), high-speed data transmission, and robust signal integrity, allowing reliable delivery of digital video signals across standard DVI cables to display devices.

The Chrontel chip manages critical video signal timing, including pixel clock generation, horizontal synchronization (HSYNC), vertical synchronization (VSYNC), and data enable (DE) signals. Correct synchronization of these signals ensures stable and flicker-free images. The chip typically supports a wide range of resolutions, accommodating various resolutions defined by FPGA configurations. Internally, the (CH7301C) also integrates modules for color space management and signal integrity control, ensuring consistent output quality and compatibility with digital displays.

The DVI standard itself is a high-speed digital interface widely used for video transmission between source devices (such as FPGA boards or graphics cards) and display monitors. DVI leverages the TMDS standard, effectively minimizing electromagnetic interference and maintaining signal integrity, making it well-suited for high-resolution, high-bandwidth digital video streams. On the ML605 FPGA board, the Chrontel (CH7301C) precisely encodes and transmits the FPGA- generated video signals, ensuring that the classification output images are clearly, accurately, and promptly displayed without data loss or distortion.

Key Components of the DVI Signal:

1) Pixel Data (RGB Values): Represents pixel colors as 8- or 10-bit RGB values. In the provided Verilog code, a 1-bit RGB representation uses the MSB for output, simplifying color processing.

2) Synchronization Signals:

- HSYNC (Horizontal Sync): Marks the start of a new pixel row.
- VSYNC (Vertical Sync): Indicates the start of a new frame, resetting the display for the next frame.

3) Clock Signal (DVI_CLK): Synchronizes pixel data and timing signals with the display's refresh rate for smooth video output.

Timing and Display Processing: The Verilog code manages timing through modules that generate pixel positions (sx, sy), frame signals, and synchronization outputs (h_sync, v_sync). These signals ensure pixel data aligns with the LCD's grid structure.

LCD Display Conversion: The LCD controller

processes incoming RGB data to adjust liquid crystal cells, determining pixel color and intensity. HSYNC, VSYNC, and DVI_CLK ensure data is applied in the correct sequence, maintaining image integrity and frame synchronization.

4.6. Displaying the Output of the Trained Neural Network on Monitor using DVI

Using the DVI to analyze trained neural network results, a system was implemented to display outputs on a monitor. This involves hardware-software integration to enable efficient and accurate visualization.

Process Overview: Neural network outputs, such as class labels or probabilities, are processed into visual formats (e.g., bounding boxes, heatmaps). A lightweight rendering engine maps these data into graphical primitives and generates images or video frames compatible with DVI displays.

DVI Signal Generation and Hardware Integration: A DVI transmitter module encodes visual content into synchronization signals (HSYNC, VSYNC) and RGB pixel data. An FPGA or microcontroller ensures compliance with DVI timing specifications, delivering high-resolution, low-latency output to the monitor

5. EXPERIMENTAL RESULTS

As a case study the proposed CNN for classification of CIFAR-10 images is implemented in Verilog as suggested in [31], to display the output classification of this network we used the proposed method, Result in Table 4. indicate the number of slice registers, slice LUTs, I/O pins and global clock buffers (BUFG) used in CNN architecture. The DVI protocol implementation on the ML605 FPGA was evaluated in terms of device utilization to highlight its resource efficiency. Our analysis focused on key FPGA resources—namely slice registers, LUTs, and logic components. Table 5. presents the synthesis report results, which reflect the resource usage of the DVI protocol module (as illustrated in Fig. 4). The notably low resource consumption (31 slice registers and 90 LUTs) underscores the lightweight nature of our design for generating synchronization signals in stage 4 explained in section IV-D.

TABLE 4
Device utilization summary for slice logic of prorosed CNN for classification of cifar-10 images

Resource	Used	Available	Utilization (%)
Slice Registers (FF)	910	301,440	0.30
Slice LUTs	1,871	150,720	1.24
I/O Pins	357	720	49.58
Global Clock Buffers (BUFG)	1	32	3.13

TABLE 5
Device utilization summary for slice logic for generating synchronization signals

Resource	Used	Available	Utilization (%)
Slice Registers	31	301,440	1
Slice LUTs	90	150,720	1
Logic Components	88	150,720	1

The results demonstrate efficient utilization of FPGA resources, with significant room for additional functionality if needed. The utilization metrics indicate that the design is optimized and suitable for real-time processing while maintaining a low resource footprint. Our experiments confirmed the success of the FPGA-based DVI protocol implementation. The results showed that the system meets DVI standards for signal synchronization and output quality, supports high-resolution video displays, and demonstrates real-time performance for image processing and displaying the predicted output prob- ability map of neural network. The following key points were confirmed through testing:

- Precise signal synchronization with no visible distortion or delay.
- High-quality video output, supporting resolutions up to 1920×1080 at 60 Hz.
- Real-time image processing with a throughput of up to 30 fps at 1280×720 resolution.
- Efficient use of FPGA resources, leaving room for additional tasks or optimizations.

These results validate the feasibility and effectiveness of using the ML605 FPGA for displaying the predicted output probability map of trained model by neural network and image processing applications and pave the way for future improvements in FPGA-based video systems. As shown in Fig. 4, the probability values of the classes in the multi-class trained neural network can be displayed on an LCD. Each class probability is represented by the width of the corresponding column bar, visually indicating the likelihood of each class.

Fig. 4 illustrates the neural network output as it is rendered on an LCD screen via the DVI protocol implemented on the FPGA. The figure shows how classification results—such as class probabilities—are translated into graphical bar elements for real-time display. Each class is represented by a colored bar, where the bar's width corresponds to the predicted probability of that class. This format enables quick interpretation of classification outcomes, supporting applications such as medical imaging or real-time object detection. The FPGA ensures precise synchronization of pixel and timing signals, allowing seamless and low-latency image rendering. This figure demonstrates the system's effectiveness in converting computational results into a clear and immediate display format, highlighting its applicability for embedded and edge AI systems.

Fig. 4. Real-time display of the CNN model's probability map on the CIFAR-10 dataset during the testing phase. The width of each color bar indicates the predicted probability of the corresponding class, with the widest bar representing the most likely classification.

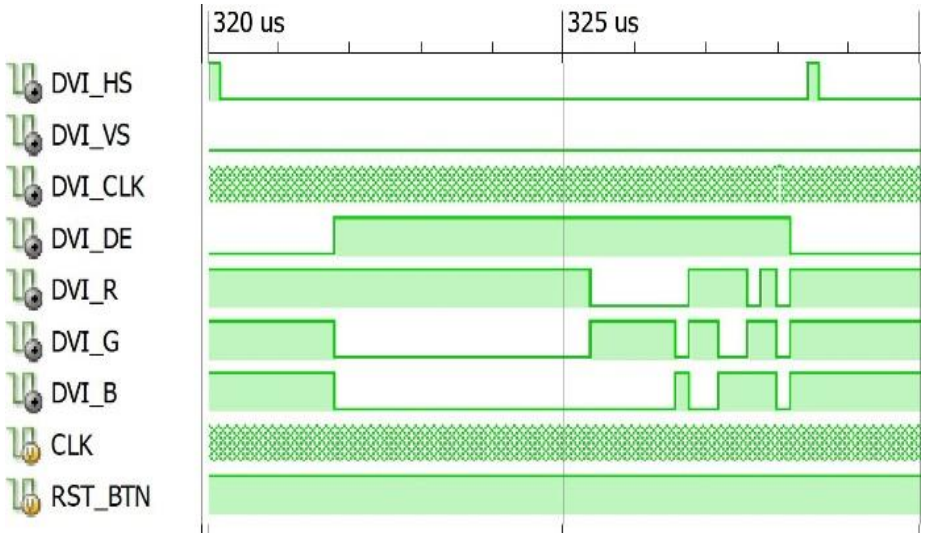


Fig. 5. Signal waveforms for timing, pixel data, and clock synchronization.

Fig. 5. shows the signal waveforms captured during the execution of the design. These waveforms illustrate the synchronization of timing signals, pixel data, and clock signals, which are critical for ensuring the correct operation of the DVI protocol.

The signal waveforms validate the proper implementation of the DVI protocol, showing stable synchronization across all required signals. These results support the design's ability to handle real-time video outputs effectively.

Using on-board LEDs to monitor the status of TMDS signals is an effective and simple method for debugging the DVI connection from an FPGA to a monitor. If the LEDs connected to the TMDS lines, such as data or clock, are not blinking, it indicates that the TMDS signals are not initialized correctly, and the issue is not related to the monitor. To troubleshoot, we ensure proper initialization of TMDS signals, verify clock settings on the FPGA, and confirm that the DVI standard configurations (e.g., resolution, sampling rate) are accurate

6. CONCLUSION

This paper presented a hardware implementation of the DVI protocol on the ML605 FPGA platform for real-time display of neural network and image processing outputs. Leveraging the capabilities of the Xilinx Virtex-6

architecture and the flexibility of FPGA-based design, a system was developed to transmit high-quality digital video signals directly to an external monitor.

The implementation addresses critical challenges such as pixel synchronization, precise timing control, and protocol compliance, thereby enabling accurate rendering of classification results without relying on external processors. The system

supports high-resolution output (up to 1920×1080 at 60 Hz) and demonstrates low resource utilization, making it suitable

for embedded and edge AI applications.

By embedding the DVI output functionality within the FPGA and integrating lightweight neural network inference, this work provides an effective hardware-software co-design framework for real-time feedback in intelligent systems. The approach is particularly valuable for tasks requiring low latency and high reliability, such as smart monitoring and medical imaging.

Future research could explore extending the design to sup- port alternative video standards (e.g., HDMI or DisplayPort), implementing more complex neural architectures, or scaling the design for multi-channel outputs. Additionally, improvements in memory access patterns, dynamic reconfiguration, or adaptive resolution could further enhance system performance and flexibility.

Overall, this work demonstrates the viability of FPGA-based systems for efficient, high-performance neural network inference and real-time display using digital video interfaces, contributing to the advancement of intelligent embedded system design.

7. REFERENCES

[1] J. Park and D. Kim. (2024, Jan.). Statistical Eye Diagrams for High-Speed Interconnects of Packages: A Review. *IEEE Access*. [Online]. *12*, pp. 22880–22891. Available: https://doi.org/10.1109/ACCESS.2024.3359037

[2] S. Singh, A. S. Mandal, C. Shekhar, and A. Vohra. (2017, Jun.). Memory Efficient VLSI Implementation of Real-Time Motion Detection System Using FPGA Platform. *Journal of imaging*. [Online]. *3(2)*, p. 20. Available: https://doi.org/10.3390/jimaging3020020.

[3] J. Plusquellic, E. E. Tsiropoulou, and C. Minwalla. (2023). Privacy-Preserving Authentication Protocols for IoT Devices Using FPGA and DVI Integration. *IEEE Transactions on Emerging Topics in Computing*. [Online]. Available: https://ece-research.unm.edu/jimp/pubs/SiRF_Authentication_FINAL.pdf

[4] H. Park and S. Kim. (2023). Overviewing AI-Dedicated Hardware for On-Device AI in Smartphones. *In Artificial Intelligence and Hardware Accelerators*. Springer International Publishing. [Online]. pp. 127–150. Available: https://doi.org/10.1007/978-3-031-22170-5_4

[5] V. Jain, S. Giraldo, J. De Roose, L. Mei, B. Boons, and M. Verhelst. (2023, Aug.). TinyVers: A Tiny Versatile System-on-Chip with State-Retentive eMRAM for ML Inference at the Extreme Edge. *IEEE Journal of Solid-State Circuits*. [Online]. *58(8)*, pp. 2360–2371. Available: https://doi.org/10.1109/JSSC.2023.3236566

[6] J. Peddie. (2023). The GPU Environment—Hardware. *In The History of the GPU—Eras and Environment*. Springer International Publishing. [Online]. pp. 151–200. Available: https://doi.org/10.1007/978-3-031-13581-1_5

[7] D. G. Bailey, "Design for Embedded Image Processing on FPGAs," John Wiley & Sons, 2023.

[8] Chrontel, Inc. (2014). *CH7301C DVI Transmitter Device Datasheet*, Rev. 2.1. [Online]. Available: CH7301C DVI Transmitter Device.

[9] M. A. Nuño-Maganda, J. H. Jiménez-Arteaga, J. H. Barron-Zambrano, Y. Hernández-Mier, J. C. Elizondo-Leal, A. Díaz-Manríquez, C. Torres-Huitzil, and S. Polanco-Martagón. (2022). Implementation and Integration of Image Processing Blocks in a Real-Time Bottle Classification System. *Scientific Reports*. [Online]. *12(1)*, p. 4868. Available: https://doi.org/10.1038/s41598-022-08777-x

[10] W. Baisi, "A Machine Learning Approach to Optimizing CNN Deployment on Tile-Based Systems-on-Chip," Doctoral dissertation, Politecnico di Torino, 2024.

[11] C. Kyrkou, C. Ttofis, and T. Theocharides. (2011, Sep.). FPGA-Accelerated Object Detection Using Edge Information. In 2011 21st International Conference on Field Programmable Logic and Application, pp. 167–170. [Online]. Available: https://doi.org/10.1109/FPL.2011.38

[12] M. Abernot, "Digital Oscillatory Neural Network Implementation on FPGA for Edge Artificial Intelligence Applications and Learning," Doctoral dissertation, Université de Montpellier, 2023.

[13] D. Guarrera, "Real-time processing of neuronal network activity measured by a high-density microelectrode matrix through an FPGA card," Thesis, Univ. of Padova, 2022.

[14] X. Kong, F. Yu, W. Yao, S. Cai, J. Zhang, and H. Lin. (2024). Memristor-Induced Hyperchaos, Multiscroll and Extreme Multistability in Fractional-Order HNN: Image Encryption and FPGA Implementation. *Neural Networks*. [Online]. 171, pp. 85–103. Available: https://doi.org/10.1016/j.neunet.2023.12.008

[15] J. W. Park, H. Lee, B. Kim, D. G. Kang, S. O. Jin, H. Kim, and H. J. Lee. (2019, Jan.). A Low-Cost and High-Throughput FPGA Implementation of the Retinex Algorithm for Real-Time Video Enhancement. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*. [Online]. *28(1)*, pp. 101–114. Available: https://doi.org/10.1109/TVLSI.2019.2936260

[16] C. Wang and Z. Luo. (2022, Oct.). A Review of the Optimal Design of Neural Networks Based on FPGA. *Applied Sciences*. [Online]. *12(21)*, p. 10771. Available: https://doi.org/10.3390/app122110771

[17] X. Liu and Y. Chen. (2023). Hybrid FPGA-GPU Co-Design for Real-Time Neural Network Inference and Visualization. In IEEE 13th Symposium on Computer Applications & Industrial Electronics (ISCAIE), pp. 712–718. [Online]. Available:

https://doi.org/10.1109/ISCAIE57739.2023.10165432

[18] V. Thyagarajan et al. (2023). Video Analytics with FPGA-Based Smart Cameras for Object Recognition in Hockey Games. Presented at International Conference on Networking and Communications (ICNWC). [Online]. Available: https://doi.org/10.1109/ICNWC57852.2023.10127549

[19] L. Nowosielski, R. Przesmycki, and M. Nowosielski. (2016, Aug.). Compromising Emanations from VGA and DVI Interface. Presented at 2016 Progress in Electromagnetic Research Symposium (PIERS), pp. 1024–1028. [Online]. Available: https://doi.org/10.1109/PIERS.2016.7734570

[20] T. T. Hoang, N. H. Nguyen, and X. T. Nguyen. (2012). A Real-Time Object-Recognition System Based on PCNN Algorithm. International Journal of Neural computing [Online]. Available: https://thuchoang90.github.io/assets/pp/2012_ICDV.pdf

[21] H. Fang, Z. Mei, A. Shrestha, Z. Zhao, Y. Li, and Q. Qiu. (2020, Nov.). Encoding, Model, and Architecture: Systematic Optimization for Spiking Neural Network in FPGAs. In Proceedings of the 39th international conference on computer-aided design, pp. 1–9. [Online]. Available: https://doi.org/10.1145/3400302.3415608

[22] C. Farabet, B. Martini, P. Akselrod, S. Talay, Y. LeCun, and E. Culurciello. (2010, May). Hardware Accelerated Convolutional Neural Networks for Synthetic Vision Systems. In Proceedings of 2010 IEEE international symposium on circuits and systems, pp. 257–260. [Online]. Available: https://doi.org/10.1109/ISCAS.2010.5537908

[23] N. Yildiz, E. Cesur, K. Kayaer, V. Tavsanoglu, and M. Alpay. (2014, Oct.). Architecture of a Fully Pipelined Real-Time Cellular Neural Network Emulator. *IEEE Transactions on Circuits and Systems I: Regular Papers*. [Online]. *62(1)*, pp. 130–138. Available: https://doi.org/10.1109/TCSI.2014.2345502

[24] K. Kayaer and V. Tavsanoglu. (2009, Apr.). A New Cellular Neural Network Emulator Architecture Processing Video Real-Time. In 2009 IEEE 17th Signal Processing and Communications Applications Conference, pp. 1024–1028. [Online]. Available: https://doi.org/10.1109/SIU.2009.5136446

[25] M. Abernot, "Digital Oscillatory Neural Network Implementation on FPGA for Edge Artificial Intelligence Applications and Learning," Ph.D. dissertation, Univ. of Montpellier, 2023.

[26] O. L. Savkay, N. Yildiz, E. Cesur, M. E. Yalcin, and V. Tavsanoglu. (2013, Sep.). Realization of Preprocessing Blocks of CNN Based CASA System on FPGA. In 2013 European Conference on Circuit Theory and Design (ECCTD), pp. 1–4. [Online]. Available: https://doi.org/10.1109/ECCTD.2013.6662238

[27] P. Antonik, *Application of FPGA to Real-Time Machine Learning: Hardware Reservoir Computers and Software Image Processing*. Springer, 2018.

[28] A. Ahilan and E. A. K. James. (2011, Dec.). Design and Implementation of Real-Time Car Theft Detection in FPGA. In Third International Conference on Advanced Computing, pp. 353–358. [Online]. Available: https://doi.org/10.1109/ICoAC.2011.6165201

[29] D. Davutoglu, N. Yildiz, U. E. Ayten, and V. Tavsanoglu. (2018). Real-Time Frame Buffer Implementation Based on External Memory Using FPGA. *Procedia Computer Science*. [Online]. *131*, pp. 641–646. Available: https://doi.org/10.1016/j.procs.2018.04.307

[30] A. Fasih, C. Schwarzlmueller, K. Kyamakya, and F. A. Machot. (2010). Video Enhancement for ADAS Systems Based on FPGA and CNN Platform. *International Journal of Signal & Image Processing*. [Online]. *1(3)*.

[31] A. Padhi and S. V. (2020). *Image-Classification-Using-CNN-on-FPGA*. [Online].

Journal of Computer and Knowledge Engineering, Vol. 8, No. 2, 2025. (25-40)

Ferdowsi University of Mashhad

Journal of Computer and Knowledge Engineering

https://cke.um.ac.ir

Information and Communication Technology Association of Iran

Smart Grid Security: Proactive Prediction of Advanced Persistent Threats*

Research Article

Motahareh Dehghan [1], Erfan.Khosravian[2]

10.22067/cke.2025.91408.1141

Abstract *The increasing reliance on Internet of Things devices in smart grids has introduced significant cybersecurity challenges, particularly in the detection and prevention of Advanced Persistent Threats. These threats, characterized by their stealth and persistence, can compromise the integrity and functionality of critical grid infrastructure. This paper proposes the use of Deep Reinforcement Learning to enhance cybersecurity in smart grids by leveraging the ProAPT model, which is specifically designed to predict and mitigate Advanced Persistent Threats. The ProAPT model utilizes a Markov Decision Process to simulate and assess potential threats, dynamically adapting to the evolving security landscape. The model is trained using the CICAPT-IIoT dataset, which includes simulated attack scenarios in industrial IoT networks. The results of our experiments demonstrate the effectiveness of the ProAPT model in detecting and preventing APTs in smart grid environments. Experimental results show that the ProAPT model significantly outperforms traditional machine learning algorithms like Random Forest, Support Vector Machines, and Logistic Regression, achieving 93.8% accuracy, 93.12% precision, 95.2% recall, and 94.15% F1-Score. The feature importance analysis reveals that traffic-related features such as packet size variance and connection duration are crucial in identifying Advanced Persistent Threats. This paper demonstrates the effectiveness of Deep Reinforcement Learning in enhancing smart grid cybersecurity by proactively identifying and mitigating cyber threats, offering a promising approach to securing IoT-based critical infrastructures against sophisticated cyberattacks.*

Key Words Cyber Security, Smart Grids, Advanced Persistent Threats, Deep Reinforcement Learning, ProAPT Model, Feature Importance.

1. INTRODUCTION

The transformation from traditional power grids to smart grids has revolutionized the energy sector by integrating modern technologies such as IoT devices, sensors, and advanced communication systems. These technologies enable real-time monitoring, automated decision-making, and predictive maintenance, making energy supply more efficient, reliable, and sustainable. In particular, smart grids enable dynamic management of electricity generation, distribution, and consumption, improving energy efficiency and facilitating the integration of renewable energy sources. However, the increasing complexity of smart grids increases their vulnerability to cybersecurity threats. The emergence of IoT in smart grids has significantly increased the number of connected devices and systems, many of which are exposed to external networks or deployed in remote or insecure environments. While these IoT devices are essential to optimizing network operations, attackers can also exploit vulnerabilities in these devices to infiltrate network systems, manipulate operations, or disrupt network operations. These threats are exacerbated by the increasing sophistication and persistence of cyber-attacks targeting critical infrastructure, which can have serious consequences such as system failure, data theft, and even property damage [1].

One of the most concerning types of cyber-attacks related to smart grids is the APT. An APT is a type of advanced, stealthy cyber-attack designed to infiltrate a network and remain undetected for long periods of time. Unlike traditional cyber-attacks, which are often short-lived and detectable by traditional defense mechanisms, APTs are characterized by their multi-stage nature and long-term objectives, making them difficult to identify and contain. These threats are often launched by well-funded and organized attackers, including nation states and cybercrime organizations, who seek to maintain persistent

* Manuscript received 2024 December 29, Revised 2025 March 4, Accepted 2025 June 16.

[1] Corresponding author. Assistant Professor Department of Industrial and Systems Engineering, Tarbiat Modares University, Tehran, Iran. **Email:** m_dehghan@modares.ac.ir

[2] Assistant Professor, Department of Mechanical Engineering, Payame Noor University, Tehran, Iran.

access to critical systems for espionage, sabotage, or data exfiltration. The impact of APTs on the smart grid is potentially devastating [2].

If attackers successfully penetrate the smart grid, they can manipulate operational data, disrupt the flow of electricity, or compromise the security of the entire system. For example, APTs could attack the power grid's control systems, causing power outages and damaging the electrical infrastructure. Furthermore, because the smart grid is decentralized and relies heavily on IoT devices for data collection and decision-making, these attacks are increasingly difficult to detect.

Traditional defense mechanisms such as signature-based IDSs and basic anomaly detection methods are often ineffective against such complex and persistent threats.
The scale and complexity of the smart grid poses unique challenges for cybersecurity. Unlike traditional IT networks, where security measures can be deployed centrally, the smart grid is comprised of numerous interconnected devices, including smart meters, grid sensors, phases of power flow management systems, and actuators. These devices are distributed across vast geographic areas and communicate with each other in real time to ensure efficient network operation. Given this dynamic and decentralized structure, ensuring the security of the smart grid requires not only protecting individual devices, but also ensuring that all components work together securely [3].

Recent research highlights the growing importance of deep learning techniques, particularly Deep Reinforcement Learning [4], in addressing the dynamic and adaptive nature of APTs in smart grids. DRL offers a promising solution for proactive cybersecurity measures by continuously learning from interactions with the environment and adapting strategies accordingly. Studies such as [5] emphasize the role of deep learning in enhancing the resilience of smart grid networks against evolving cyber threats. In addition, Sewak et al. [6] demonstrate the effectiveness of DRL-based models in detecting complex cybersecurity threats, including APTs, by using reward-based learning frameworks. These advancements are particularly relevant for smart grid systems, where traditional cybersecurity measures are increasingly inadequate due to the rapid evolution of attack techniques and the scale of connected devices.

Moreover, recent studies such as [7,8] have proposed robust models integrating machine learning and DRL for detecting and mitigating APTs. They have designed a DRL framework for smart grid cybersecurity, highlighting its ability to adapt to the complex, dynamic nature of cyber-physical attacks. Khan et al. [7] provide an overview of the cyber threats facing modern smart grids and propose advanced machine learning models to counter these challenges. These studies reinforce the need for adaptive and proactive cybersecurity frameworks like the ProAPT model, which utilizes DRL to predict and mitigate APTs before they fully manifest, thus improving the security and reliability of smart grids. In addition, IoT devices often have limited processing power and storage capacity and may not support traditional security measures, further complicating the detection and containment of complex cyber threats. Furthermore, the growing reliance on M2M communications and cloud computing in smart grids increases the attack surface and provides attackers with numerous entry points. This is particularly problematic because attackers may exploit vulnerabilities in the software or hardware of IoT devices, as well as in communication protocols and network interfaces.

As a result, traditional cybersecurity approaches are no longer sufficient to address emerging threats to smart grids. Given the limitations of traditional techniques and the increasing sophistication of cyber-attacks, there is an urgent need for more advanced and adaptive solutions that can effectively detect, predict, and mitigate APTs in smart grids [9].

In this paper, we propose a novel solution to combat cybersecurity threats in smart grids by detecting and mitigating APTs using DRL. DRL is a branch of machine learning in which an agent learns how to make optimal decisions by interacting with the environment and receiving feedback in the form of rewards or penalties [10]. Unlike supervised learning approaches that require labeled data, DRL operates in dynamic environments and is able to continuously learn from new interactions and adapt its strategy accordingly. The proposed solution leverages the ProAPT (Prediction of Advanced Persistent Threats) model [11], which is designed to predict and mitigate APTs using deep reinforcement learning. The central idea behind the ProAPT model is to use a Markov decision process (MDP) to simulate the evolving security state of a smart grid system and determine the optimal action to address potential threats.

In the context of a smart grid, these actions might include triggering security protocols, isolating affected devices, or adjusting network configurations to prevent the attack from spreading. By continually interacting with the grid’s environment and receiving feedback, the model learns how to improve threat detection and mitigation strategies over time, enabling it to identify APTs before they fully manifest. One key innovation of this approach is its ability to proactively predict APTs. Rather than relying on reactive measures such as post-attack detection, the ProAPT model predicts possible future threats based on historical attack data and ongoing grid activity. This proactive approach significantly improves the resilience of the grid, enabling early intervention to prevent severe damage. The model is trained using the CICAPT-IIoT dataset [12], which contains simulated attack scenarios in industrial IoT networks. The ProAPT model is applied to this dataset to evaluate its effectiveness in detecting and mitigating APTs in smart grid environments.

This paper makes the following key contributions to advancing smart grid cybersecurity:

- Novel Application of DRL for APT Prediction: Unlike previous works that rely on traditional machine learning approaches, this study pioneers the use of DRL to predict APTs in smart grids, enabling a more adaptive and proactive defense mechanism.
- Empirical Validation on a Real-World Industrial IoT Dataset: We rigorously evaluate our proposed ProAPT model using the CICAPT-IIoT

dataset, which includes diverse and realistic cyber-attack scenarios specific to critical infrastructure, ensuring practical relevance and generalizability.

- Feature Importance-Driven Model Optimization: Our approach integrates a feature importance analysis to systematically identify and prioritize the most critical features, enhancing model interpretability and efficiency.
- Comprehensive Performance Assessment: Unlike prior studies that focus on limited evaluation metrics, we conduct an extensive performance analysis using accuracy, precision, recall, and F1-score to provide a holistic understanding of the model's effectiveness in detecting and mitigating APTs.

This paper is structured as follows:

Section 2 provides a comprehensive overview of related research in the areas of cybersecurity in smart grids, APT detection, and the application of DRL in cybersecurity. Section 3 presents the methodology detailing the ProAPT model, its adaptation to smart grid cybersecurity, the training process and evaluation using the CICAPT-IIoT dataset. Section 4 describes the experimental setup including the results of applying the ProAPT model and compares its performance with other conventional models in terms of detection accuracy, precision and F1-score. In section 5 feature importance methods are implemented and the best features are stated. Finally in sections 6 and 7 we discuss and conclude the paper with an overview of the contributions and suggestions for future work such as improving scalability and integrating it into existing smart grid security frameworks.

2. RELATED WORK

Smart grid cybersecurity is a critical concern due to the integration of advanced technologies and data-driven systems, which, while enhancing efficiency and sustainability, also introduce vulnerabilities. These vulnerabilities manifest in various forms, such as false data injection attacks, malware, and cyber-physical attacks, posing significant risks to the integrity and reliability of smart grids. Addressing these threats requires a multifaceted approach involving detection, prevention, and mitigation strategies. Machine learning models, such as Extra Tree, Random Forest, and Extreme Gradient Boosting, have shown high accuracy (up to 98%) in detecting these attacks, providing a robust defense mechanism [13].

Cyber-Physical attacks involve manipulating power demands using IoT devices or introducing false sensor readings. A DRL framework has been proposed to counter these attacks by triggering appropriate protection sequences, verified through reachability analysis for safety [14]. The use of SCADA systems in smart grids makes them susceptible to malware, which can exploit IT-OT integration vulnerabilities. The complexity of these systems increases the risk of cyber threats, necessitating enhanced cybersecurity measures [15].

The integration of information and operations technology in smart grids introduces new vulnerabilities, requiring continuous monitoring and updating of security protocols to prevent breaches [16]. Implementing a combination of traditional and advanced security measures is crucial. This includes regular updates, intrusion detection systems, and employee training to recognize and respond to threats [7]. Ongoing research is essential to address emerging threats and develop innovative solutions, such as advanced algorithms for attack detection and mitigation [16]. While smart grids offer numerous benefits, such as improved energy efficiency and integration of renewable sources, they also present unique cybersecurity challenges. The dynamic nature of cyber threats necessitates a proactive and adaptive approach to security, ensuring the resilience and reliability of smart grid infrastructures.

Smart grids are an essential part of modern energy systems, but they are also vulnerable to various cybersecurity threats due to their increasing reliance on digital technologies and interconnected devices. Researchers have proposed several solutions to secure smart grids, which can be broadly categorized into IDS, anomaly detection techniques, and authentication protocols. One of the primary methods used to protect smart grids is the development of IDS, which monitor the network for any signs of unauthorized access or abnormal behavior. IDS in smart grids often rely on signature-based detection, which matches observed network behavior to known attack patterns. However, as smart grid environments evolve, this approach has become less effective due to the increasing sophistication of cyberattacks and the dynamic nature of smart grids. To address this limitation, anomaly detection techniques, such as statistical methods and machine learning, have been integrated into IDS to detect deviations from normal operations that could indicate a security breach [17].

These methods, though effective in detecting new types of attacks, struggle with issues such as false positives and the need for large amounts of labeled data. Detecting and responding to APTs in smart grids presents unique challenges. APTs are characterized by their stealthy, multi-stage nature and ability to remain undetected over long periods. This makes them particularly dangerous in smart grids, where attackers can potentially gain control of critical infrastructure systems without alerting security systems. Additionally, the heterogeneity of smart grid components, the presence of many IoT devices, and the distributed nature of control make it difficult to monitor and secure the entire grid effectively. These challenges require advanced, dynamic methods of detection and response that can adapt to new and evolving attack vectors. Several studies have explored using real-time monitoring and adaptive security models to mitigate these challenges [5].

APTs are one of the most critical cybersecurity concerns for modern infrastructure, including smart grids. Unlike typical cyberattacks, which tend to be short-lived and easily detectable, APTs are long-term attacks that exploit vulnerabilities in a system over an extended period. APTs often involve multiple stages, including initial infiltration, lateral movement within the network, data exfiltration, and

maintaining persistence over time. They are designed to avoid detection and maximize their impact on targeted systems [2].

APTs are usually associated with highly organized threat actors, such as nation-states or cybercriminal groups. These actors have significant resources and expertise, allowing them to plan and execute multi-phase attacks. Key characteristics of APTs include sophistication, long-term persistence, and specific targeting. APT attacks often target high-value assets, including critical infrastructure like power plants, water supplies, and transportation systems, with the goal of gaining unauthorized access, stealing sensitive data, or causing operational disruptions [18].

Some of the most infamous APT attacks targeting critical infrastructure include Stuxnet [19], which specifically targeted Iran's nuclear facilities, and BlackEnergy [20], which affected Ukraine's power grid. These attacks demonstrate the high stakes involved in cybersecurity for critical infrastructure and the potential consequences of a successful APT. Stuxnet, for example, manipulated control systems within the targeted facility, leading to significant physical damage. Traditional methods for detecting APTs include signature-based approaches, which compare network traffic to predefined attack patterns, and statistical methods, which look for anomalies in system behavior that may indicate an attack. However, these approaches often struggle to detect sophisticated, low-and-slow APTs. Recent research has focused on leveraging machine learning techniques to improve APT detection. Models such as random forests, support vector machines (SVM), and deep learning have shown promise in identifying previously unknown attack patterns. Despite this progress, a major challenge remains the lack of labeled data for training models, as APTs are rare and difficult to simulate in a controlled environment [21].

DRL has emerged as a powerful tool for addressing complex decision-making problems in dynamic environments, including cybersecurity. DRL involves training an agent to take actions in an environment to maximize cumulative rewards, making it an ideal approach for security tasks that require continuous adaptation and learning. DRL has shown great potential in the field of cybersecurity due to its ability to adapt to evolving threats and optimize long-term security strategies. DRL-based models have been used for tasks such as intrusion detection, vulnerability scanning, attack detection, and incident response. By continuously learning from the environment and adjusting its actions based on feedback, DRL can provide an adaptive, proactive defense mechanism against cyberattacks, including APTs. For example, DRL has been used to model intrusion detection in IoT networks, where it learns to distinguish between benign and malicious activities based on observed behaviors [22].

One of the main advantages of DRL is its ability to learn optimal decision policies from raw data without relying on hand-crafted rules or predefined attack signatures. This capability is particularly useful in environments like smart grids, where attack patterns are constantly evolving. Moreover, DRL-based models can handle complex, multi-step security tasks that require dynamic adjustments based on the state of the system. For instance, DRL can optimize actions to prevent attacks while minimizing the impact on system performance and resource consumption. Despite its potential, applying DRL to cybersecurity poses several challenges. One of the main challenges is the sample inefficiency of deep reinforcement learning algorithms, where a large number of interactions with the environment are often needed to converge on an optimal policy. Additionally, reward shaping can be difficult, as determining the appropriate rewards for specific security actions in dynamic environments like smart grids is not straightforward. Finally, training DRL models in real-world cybersecurity scenarios often requires access to large amounts of labeled data, which is typically not available for rare events such as APTs [6].

DRL, a promising technique for cybersecurity, enables models to learn optimal responses by interacting with the environment and adapting over time. It has shown significant potential in various fields, including robotics, gaming, and cybersecurity. One notable application is ProAPT [11], which uses DRL to predict the next stages of APTs. The model learns from historical attack data and environmental conditions to anticipate the next steps in an ongoing attack, enabling proactive defense mechanisms.

Recent advances in DRL have led to a surge of research focused on enhancing cybersecurity in smart grids and critical infrastructures. Abdi et al. [5] provided a comprehensive survey on the application of deep learning, particularly DRL, to proactively secure smart grid environments. They emphasized how DRL frameworks can adaptively counter zero-day attacks and sophisticated APTs. Veith et al. [23] explored how DRL agents trained on misuse cases can learn novel attack vectors, representing a significant leap in proactive APT detection. Sinha et al. [24] extended this work by proposing a cyber-resilient demand response system, which not only optimizes grid operations but also integrates DRL for enhanced security against APTs and false data injection. Furthermore, Li et al. [25] introduced a state-adversarial DRL-based scheduler for integrated energy systems that mitigates the effect of data manipulation attacks on demand-response coordination. To support secure communication in grid CPS, Sun et al. [26] proposed a DRL-based multi-agent scheme for secure resource allocation under adversarial conditions.

These contributions collectively reinforce the relevance and applicability of DRL—especially DQN variants—in detecting and mitigating APTs across multiple smart grid environments [27]. While the previous research demonstrate important progress in applying machine learning and deep learning methods to smart grid cybersecurity, several critical gaps remain that hinder their real-world applicability. Most of the existing deep learning models—such as LSTM, CNN, and GRU—operate in a supervised learning setting and rely heavily on large volumes of labeled data. This is a significant limitation in the context of APTs, which are rare, highly complex, and difficult to label accurately due to their stealthy and evolving nature. Moreover, many previous solutions are static in their behavior and lack the ability to adapt over time. As cyber threats in smart grid environments grow more dynamic, fixed models trained on historical data may

struggle to detect novel attack strategies. Another notable limitation is the frequent separation between different data modalities. Prior studies often focus on either network traffic or system behavior independently, rather than combining both for richer context-aware detection. The proposed ProAPT model addresses these limitations through its integration of deep reinforcement learning with LSTM-based temporal modeling, allowing it to dynamically learn and predict sequential attack stages. Unlike static models, ProAPT can adapt to new patterns without requiring manual retraining.

3. METHODOLOGY

The ProAPT model [11] is a novel approach designed for predicting and mitigating APTs using DRL**.** The model leverages DRL's ability to continuously learn and adapt to dynamic environments, making it ideal for addressing the evolving nature of cyber threats in complex systems like smart grids. Smart grids present unique challenges due to their complexity, scale, and reliance on interconnected IoT devices.

The ProAPT model is based on Q-learning and LSTM to project the following step of APTs. As some relations exist between the attack steps, LSTM is used for value function approximation. LSTM is a modified version of RNN and facilitates the recall of past data and solves the problems of RNN. LSTM is employed to keep the previous states over long periods. The APT projection problem can be considered as a Markov Decision Process. Detection of normal or abnormal behavior at the current time step will alter the environment. The changing environment will also influence the next decision. Hence, it is natural to adapt this problem to the framework of Reinforcement Learning. We describe the Deep Reinforcement Learning System for the APT projection problem as follows: We demonstrate each state by features such as the source IP address, destination IP address, source port number, destination port number, timestamp, attack type, header length, flow duration. The agent receives the current state and selects the best action based on the ϵ -greedy policy. Indeed, the agent receives the correlated alerts and selects the following attack step. The reward is 1 or 0 for a correct/incorrect attack prediction. We use a Q-learning algorithm to learn the agent. To approximate the Q function, we employ LSTM, as some relations exist between attack steps. A Q function provides the maximum expected reward at a specific state and action. We employ APT datasets instead of interacting with the environment to reduce the time spent learning, testing, and evaluating. Although employing datasets increases the speed of learning and testing, interacting with the environment is suitable for predicting unknown APTs.

As mentioned, we give data from an APT dataset as input to DRLS. Based on the input data, the agent learns how to predict the following step of attacks.

Based on Fig. 1, we randomly divide the input dataset into sections and select the index. Then, from the selected index, we consider N number of data as training data. Each Training data, as input for LSTM, include the features of the alerts such as source IP address, destination IP address, source port number, destination port number, timestamp, attack type, header length, and flow duration. The second part is the data label in step t+1. This part shows the attack label in step t+1 such as automated collection, screen capture, exfiltration over C2 channel, ingress tool transfer.

For example, S_0 represents the attack step at (t_0), and a_1^* expresses the attack label at time (t_1) and for the state S_1. Since we want to recognize the following step of the attack in the DRLS, we consider the following step label in each state and use it to determine its reward. Fig. 2 demonstrates a DRLS to predict the following attack step. As mentioned, we give data from an APT dataset as input to DRLS. Based on the input data, the agent learns how to predict the following step of attacks. Input data consists of three parts. The first part expresses the state at time (t). This part includes the features of the correlated alerts at time (t). The second part is the data label in step (t+1). This part shows the attack label in step (t+1). The third part describes the state at time (t+1). That is a feature of correlated alerts at time (t+1). The first part of the input is entered into the LSTM neural network to approximate the value function of different actions for the state at time (t). In this context, LSTM approximates the value function for the following step of the ongoing attack. We display the approximated value with ($\hat{a_{t+1}}$), which has the value ($\hat{q_{t+1}}$). Then, based on the ϵ -greedy policy, the action with the highest value function is selected by a probability of ϵ . Finally, the approximated value ($\hat{a_{t+1}}$) is compared with the main label of the following step of the attack, which is the second part of the input data (a_{t+1}^*). If the comparison result is equal, the reward (+1) is given to the agent; otherwise, the reward (0) is given. The third part of the input is used to calculate the error function and update the LSTM. So that the state-expressing features at time (t+1) are entered in the second LSTM for approximation of value functions for different actions. At this point, the policy is the selection of action with the most significant value. In our problem, actions are the following step of attacks. Then, the obtained value is used to calculate the Mean Square Error Loss between the Q-value approximated by the LSTM for the state at time t and a reference value (q_{ref}). The reference value ($q_{ref} = r_t + \lambda \times \hat{q_{t+1}}$.) is obtained by adding the reward at time t (r_t) to the Q-value for the state at time t+1 multiplied by a discount factor (λ). The pseudocode for the smart grids APT prediction is depicted in Algorithm 1.

The output space (actions) corresponds to predicting the next attack step in an APT sequence is stated in Table 1.

Algorithm 1. Smart Grids APT prediction

Initialize Replay Buffer B
Initialize Q-network with LSTM: Q(s, a; θ)
Initialize Target Q-network with parameters θ− ← θ
Set discount factor γ and exploration rate ε

for episode = 1 to MaxEpisodes:
  Initialize state s_0 using features of alert at time t_0
  for each step t in episode:
    # Step 1: Choose action using epsilon-greedy policy
    if random() < ε:
      Select random action a_t (i.e., randomly select next predicted attack step)
    else:
      Select action a_t = argmax Q(s_t,a; θ)

    # Step 2: Perform action (predict next step)
    Observe reward r_t (1 if correct prediction, 0 otherwise)
    Observe new state s_{t+1} from alert features at time t_{+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in B

    # Step 3: Sample mini-batch of transitions from B
    for each sampled (s_i, a_i, r_i, s_{i+1}):
      Predict target value:
      Q_target = r_i + γ * max_a' Q_target(s_{i+1}, a'; θ−)
      Update Q-network using MSE loss:
        Loss = (Q(s_i, a; θ) - Q_target)²

    Every C steps:
      θ− ← θ # Update target network

    $s_t \leftarrow s_{t+1}$

  Decay ε (exploration rate)

- State (s_t): Vector of features at time t such as source IP, destination IP, source port, destination port, timestamp, attack type, header length, flow duration
- Action (a_t): The predicted next attack step from a predefined set of APT steps derived from the MITRE ATT&CK framework.

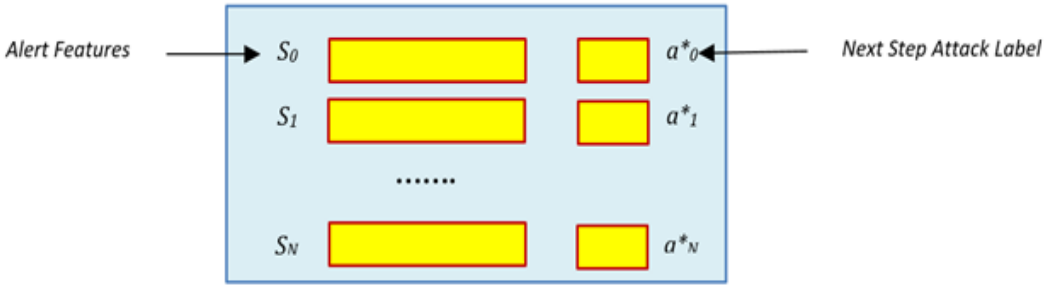


Fig. 1. Data Preparation in ProAPT model (Dehghan et al., 2022)

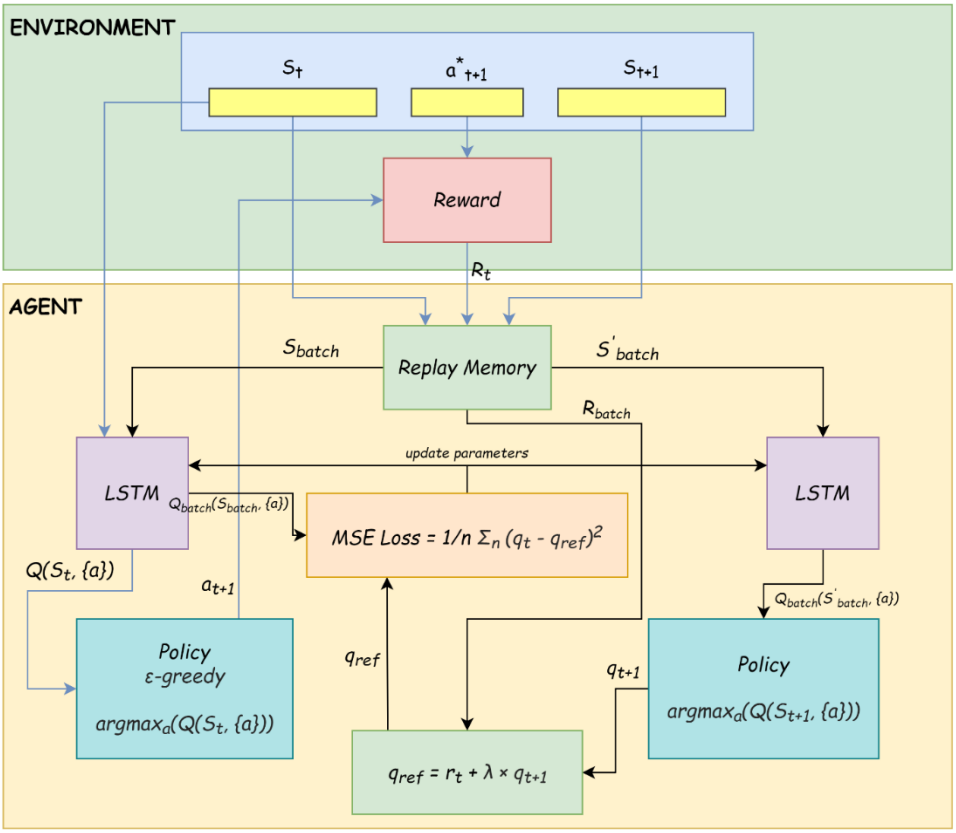


Fig. 2. The architecture of the ProAPT Model (Dehghan et al., 2022)

TABLE 1
The output space corresponds to APT prediction

Action ID	Predicted Attack Step	Description
0	Automated Collection	Data staged for exfiltration
1	Screen Capture	Attacker takes screenshots
2	Exfiltration over C2 Channel	Sensitive data exfiltrated using covert communication
3	Ingress Tool Transfer	Uploading malicious tools for further exploitation
4	Credential Dumping	Extraction of credentials from memory or files
5	Remote SSH	Remote access for lateral movement
6	Masquerading	Use of deceptive filenames/paths to evade detection
7	Data Destruction	Deletion of logs or sabotage
...	(Additional tactics as needed)	Aligned with MITRE categories from the CICAPT-IIoT dataset

Fig. 3. Pre-processing Steps

The steps of our methodology is as follows:

Data Preprocessing: Initially, the data is preprocessed to standardize features and address any missing values, as shown in Fig. 3.

Hyperparameters Tuning: Key hyperparameters like learning rate, discount factor, and exploration rate are carefully tuned to maximize the model's performance. Grid search is used to find the optimal combination of these hyperparameters, ensuring the model performs at its best.

Feature Selection: Performed using Random Forest-based feature importance to select the most informative attributes.

Data Splitting: The dataset is split into 70% for training, 30% for testing

Model Training: The model undergoes training using the reinforcement learning framework. As it trains, the model updates its Q-values based on the feedback it receives from the reward function, gradually refining its predictions over time.

Test and Evaluation: Once trained, the model is evaluated using standard classification metrics like accuracy, precision, recall, F1-score. These metrics gauge how well the model predicts the next attack in the sequence, considering both correct predictions and penalties for mistakes. By evaluating the model with these metrics, we can assess its effectiveness in predicting the next step in an attack sequence and its overall value in enhancing smart grid cybersecurity with proactive defense strategies.

4. EXPERIMENTS AND RESULTS

The CICAPT-IIoT dataset [12] is employed to evaluate the proposed prediction model. This dataset is designed for cybersecurity research, specifically for detecting APTs in industrial Internet of Things environments. The dataset simulates a sophisticated APT campaign based on the APT29 attack group, capturing both provenance logs and network traffic data from a hybrid testbed that integrates real and simulated IIoT components. The CICAPT-IIoT dataset was generated using a controlled IIoT testbed built on the Brown-IIoTbed framework, featuring a combination of physical and virtual components. It consists of two main data types: provenance logs, and network traffic logs.

The provenance logs capture system-level interactions and process relationships through a provenance graph. It includes 32 unique features, tracking process execution, file access, and network connections. The network traffic logs include an attack information file, detailing attack timestamps, process IDs of malicious actions, and attack categories, enabling researchers to correlate network activity with specific APT tactics. This dataset realistically replicates multi-stage APT campaigns relevant to smart grid cybersecurity. The attack framework follows MITRE ATT&CK tactics, encompassing over 20 distinct attack techniques across eight major categories as stated in Table 2. [12].

The dataset's attack scenarios closely mimic real-world threats to smart grids, where attackers exploit vulnerabilities in IIoT devices, industrial control systems, and network infrastructure. By incorporating provenance-based monitoring and network traffic analysis, this dataset provides a robust foundation for machine learning-based APT detection in critical infrastructure security.

As stated above, the dataset used for this research is the CICAPT-IIoT dataset, which provides a rich set of features related to the operation of Industrial Internet of Things (IIoT) devices and the detection of network-based threats, including. This dataset includes real-time network traffic data, device status, and attack patterns, which serve as the input for our DQN and LSTM models. To train the ProAPT model, we preprocess the dataset following the steps outlined in Fig. 3 Next, we select the best hyperparameters. Table 3. demonstrates the best selected one.

By setting a low learning rate, we ensure that the updates to the model remain stable. Additionally, a high discount factor emphasizes long-term rewards, helping the model prioritize future outcomes. A low exploration rate encourages the model to exploit the policies it has already learned, while a larger batch size and higher update frequency help stabilize the training process.

TABLE 2
Attack Techniques Used in CICAPT-IIoT Dataset [12]

Tactic	**Example Techniques**	**Relevant APT Groups**
Collection	Data Staging, Screen Capture	APT28, APT29, APT39
Exfiltration	Exfiltration over C2 Channels	Lazarus, APT3, APT32
Command & Control	Ingress Tool Transfer	APT29, APT3
Persistence	Event-Triggered Execution	APT28, APT29, APT3
Discovery	System & Network Discovery	Chimera, Dragonfly, APT29
Credential Access	Unsecured Credentials, Password Extraction	APT3, APT39, HEXANE
Lateral Movement	Remote SSH Access	APT29, Lazarus
Defense Evasion	Masquerading, Data Destruction	APT28, APT29, Dragonfly

TABLE 3
The best hyperparameters

Model	Hyperparameter	Value
DQN with LSTM	Action Space	Security measures (e.g., block traffic, adjust security policies)
	State Space	Network traffic features, device status, attack signatures
	Neural Network Architecture	Fully connected feedforward network
	Learning Rate	0.001
	Replay Buffer Size	10,000
	Batch Size	64
	Epsilon (for exploration)	1 (decaying to 0.1)
	Target Network Update Frequency	Every 100 steps
	Input	Sequences of time-series data (traffic, device status)
	Number of LSTM Units	100
	Learning Rate	0.001
	Epochs	50
	Batch Size	64
	Activation Function	ReLU (hidden layers), Softmax (output)

To evaluate the performance of our prediction model, we use several key metrics, as outlined by Carvalho et al. [28]:

Accuracy: This measures the proportion of correct predictions out of all predictions. It provides an overall indication of how well the model is performing in predicting the next attack step in the sequence.

Precision: Precision assesses how many of the predicted attacks are actually correct. This is particularly important in cybersecurity, as false positives can have significant consequences. A high precision ensures that the model isn't falsely predicting attack steps.

Recall: Recall measures how many of the actual attacks were correctly predicted. In cybersecurity, this metric is crucial because we want to make sure the model doesn't miss any attacks, even if it leads to a few false positives.

F1-Score: The F1-score is the harmonic mean of precision and recall, providing a balanced measure of both. It is especially valuable when dealing with imbalanced datasets, such as when attacks are less frequent than normal behavior.

Time Consumption (ms): The amount of time each model takes to process the data and make predictions. More complex models like DQN typically take longer to process due to their deeper architectures and the need for more computations.

Bandwidth Usage (KB/s): The amount of bandwidth consumed during data transfer between the model and the system. Models that require processing more complex data often use more bandwidth due to the need for transmitting larger volumes of information.

Throughput (ops/s): The number of operations the model can perform per second. Models with optimized architectures and faster computation capabilities generally have higher throughput, meaning they can handle more operations in a shorter amount of time

These metrics are essential for assessing how well the model can predict the next steps in a multi-step attack sequence. In particular, precision and recall are crucial in cybersecurity to minimize false positives and ensure that attacks are detected in a timely manner [29]. We compared the proposed ProAPT model with additional deep learning (non-reinforcement) baselines beyond traditional ML models. Specifically, we included models widely used in temporal classification tasks such as GRU, Bi-LSTM, CNN- LSTM, and Transformer architectures, as depicted in Table 4. These models were trained on the same CICAPT-IIoT dataset and evaluated using the same metrics as ProAPT to ensure a fair comparison.

As shown in Table 4, ProAPT achieved the best accuracy and F1-score but required slightly more processing time and bandwidth compared to simpler models like GRU and LSTM. However, its ability to handle complex multi-stage attack sequences and maintain high throughput demonstrates its suitability for real-time cybersecurity in smart grid systems.

To select the most suitable deep reinforcement learning algorithm for the proposed model and the dataset, we evaluated various algorithms (DQN, Double DQN, PPO, A3C), among which DQN delivered the best results. A comparison of these algorithms is presented in Table 5.

These algorithms are widely used in complex reinforcement learning environments due to their stability and robustness in continuous and asynchronous settings. PPO employs a clipped objective function to maintain policy updates within a trust region, improving learning stability. A3C, on the other hand, leverages multiple asynchronous agents to stabilize training and efficiently explore large state spaces [30].

We implemented PPO and A3C using the same environment setup, state space, and reward functions used for DQN and Double DQN to ensure consistency. Our

results, summarized in Table 5, show that while both PPO and A3C performed competitively, the proposed DQN-based ProAPT model outperformed them in terms of accuracy, precision, and recall. Specifically, PPO. These results reinforce the suitability of DQN for discrete action spaces typical of smart grid security environments, where decisions like blocking IPs or raising alarms are categorical in nature. Moreover, we considered a hybrid model combining feature-engineered inputs with a lightweight anomaly detection layer before feeding into DRL. Although this hybrid approach improved interpretability slightly, it did not outperform the standalone DRL models in overall metrics. These additional comparisons support our choice of DQN as a highly effective and practical baseline for APT detection in smart grid environments, while also highlighting avenues for future exploration in combining DRL with hybrid or ensemble methods [31].

TABLE 4
Performance Comparison between ProAPT and Deep Learning Models

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Time (ms)	Bandwidth (KB/s)	Throughput (ops/s)
ProAPT (DQN + LSTM)	92.5	91.8	93.2	92.5	150	120	5000
LSTM	89.8	88.4	91.0	89.7	95	90	5800
GRU	89.3	88.1	90.4	89.2	87	85	5900
Bi-LSTM	90.2	89.6	91.3	90.4	110	100	5600
CNN-LSTM	90.7	89.8	92.0	90.9	125	105	5400
Transformer	91.0	90.5	92.2	91.3	140	115	5100

TABLE 5
Performance Comparison between DQN, and Other DRL Models

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Time Consumption (ms)	Bandwidth Usage (KB/s)	Throughput (ops/s)
DQN	92.5	91.8	93.2	92.5	150	120	5000
Double DQN	90.3	89.5	91.4	90.4	160	115	4800
PPO	91.0	90.2	92.1	91.1	180	110	4700
A3C	88.2	86.9	89.4	88.1	200	100	4500
Hybrid	90.1	89.0	90.2	89.6	250	130	4700

5. FEATURE IMPORTANCE

Feature importance indicates how much each feature contributes to the predictions made by a machine learning model. In the case of the Random Forest Classifier, the importance of each feature is determined by its ability to reduce uncertainty or enhance decision-making at each split in the trees [32]. Decision trees within the Random Forest algorithm aim to find patterns in the data that best separate the different classes, such as benign behavior and various types of attacks (DoS, etc.). Features that result in the most impactful splits—those that effectively distinguish between these classes—are considered more important. Fig. 4 provides an overview of the feature importance results.

For the Network Traffic dataset, features related to traffic patterns, such as packet size variance, connection duration, and protocol usage, dominated the importance scores. Features indicating irregularities in network flow (e.g., unusually large data packets or abrupt connection terminations) were highly predictive of threats. Certain features, like general connection metadata, showed low importance and could potentially be excluded to streamline model training. The leading features include packet size variance, connection duration, and frequency of specific protocols. This highlights that deviations in normal traffic patterns and protocol behaviors are indicative of advanced persistent threats. After implementing feature importance, we train and test the model, and summarize the results as demonstrated in Table 6.

- The confusion matrix for multi-stage attacks before feature selection is presented in Table 7. and Fig. 5. This matrix shows the actual vs predicted values for each of the 7 attack classes. The TP (True Positives), FP (False Positives), FN (False Negatives), and TN (True Negatives) for Class 0 (as an example) are calculated as follows:
- True Positives (TP): 10000 (correctly classified instances of Class 0).
- False Positives (FP): 1250 (the number of instances from other classes that were misclassified as Class 0).
- False Negatives (FN): 860 (the number of instances of Class 0 that were incorrectly classified into other classes).
- True Negatives (TN): 2084650 (all other instances not related to Class 0).

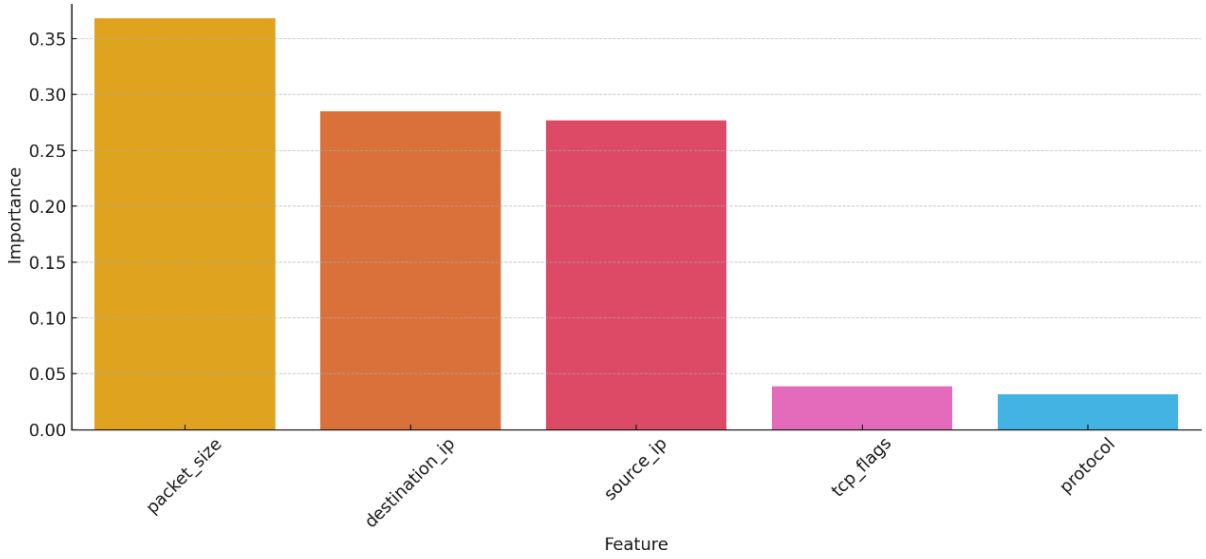


Fig. 4. Feature Importance Result for Network Traffic Dataset using Random Forest

TABLE 6
The results of prediction after feature importance implementation

Metric	Value
Accuracy (%)	93.8
Precision (%)	93.12
Recall (%)	95.2
F1-Score (%)	94.15
Time Consumption (ms)	150
Bandwidth Usage (kb/s)	120
Throughput (ops/s)	5000

TABLE 7
Confusion matrix before feature selection

	Predicted Class 0	**Predicted Class 1**	**Predicted Class 2**	**Predicted Class 3**	**Predicted Class 4**	**Predicted Class 5**	**Predicted Class 6**
Actual Class 0	10000	500	300	200	100	50	100
Actual Class 1	400	9500	400	300	200	100	150
Actual Class 2	200	300	9600	500	300	200	150
Actual Class 3	100	150	300	9700	400	300	200
Actual Class 4	50	100	200	400	9600	500	300
Actual Class 5	30	60	100	200	350	9800	500
Actual Class 6	80	120	150	300	400	450	9500

Moreover, the confusion matrix for multi-stage attacks after feature selection is presented in Table 8. and Fig. 5. After feature selection, the TP, FP, FN, and TN for Class 0 (as an example) are recalculated:

1) True Positives (TP): 10500 (correctly classified instances of Class 0).
2) False Positives (FP): 1100 (the number of instances from other classes that were misclassified as Class 0).
3) False Negatives (FN): 800 (the number of instances of Class 0 that were incorrectly classified into other classes).
4) True Negatives (TN): 2086250 (all other instances not related to Class 0).

Table 9. presents a comparison of the proposed ProAPT model with several recent works in the field of APT detection in smart grids. This comparison includes evaluation metrics such as accuracy, precision, recall, and F1-score, as well as important factors like the method used, dataset, attack types, and the year of publication. The selected works focus on applying deep learning techniques and machine learning methods to address cybersecurity threats in smart grids and IoT environments.

TABLE 8
Confusion Matrix after Feature Selection

	Predicted Class 0	**Predicted Class 1**	**Predicted Class 2**	**Predicted Class 3**	**Predicted Class 4**	**Predicted Class 5**	**Predicted Class 6**
Actual Class 0	10500	400	250	150	50	30	50
Actual Class 1	350	9800	350	250	150	80	100
Actual Class 2	150	250	9800	400	250	150	100
Actual Class 3	50	100	250	9800	300	250	150
Actual Class 4	30	60	150	300	9700	400	250
Actual Class 5	20	50	80	150	300	9700	400
Actual Class 6	60	100	120	250	350	400	9700

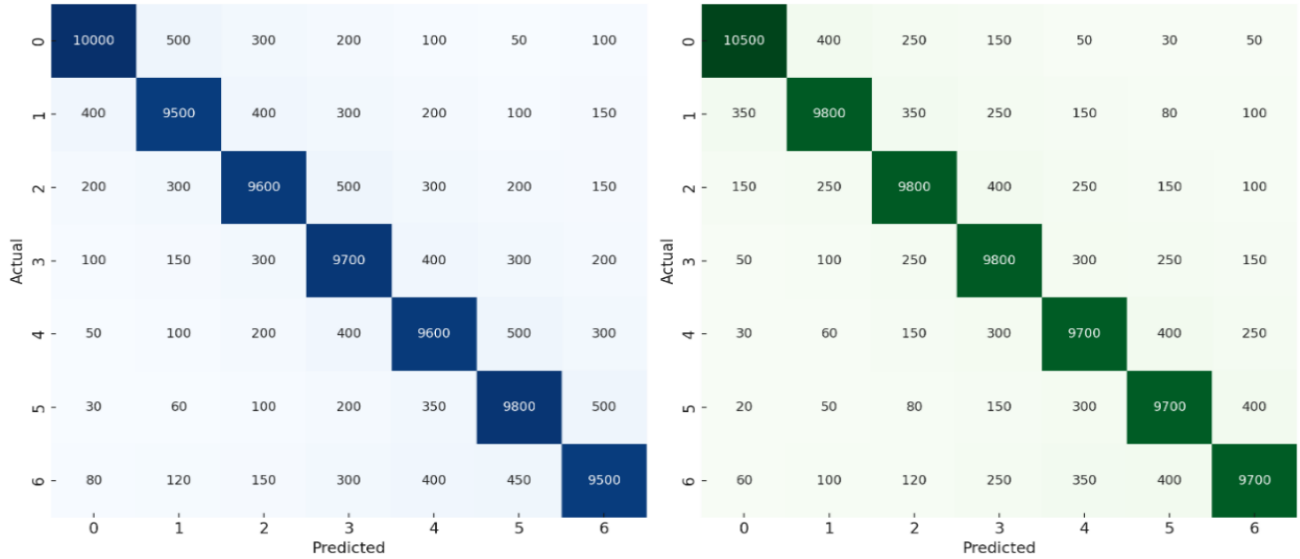


Fig. 5. Confusion Matrix before Feature Selection (Blue Diagram) and after Feature Selection (Green Diagram)

TABLE 9
Comparison of the Proposed Method with Recent Works

Study/Model	**Method**	**Dataset**	**Attack Types**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**
ProAPT (DQN) [11]	Deep Reinforcement Learning (DQN)	CICAPT-IIoT (2024)	APTs	93.8	93.12	95.2
Abdi et al. (2024) [5]	Deep Learning	Smart Grid Dataset	Malware, DoS, DDoS	90.0	89.5	91.0
Maiti & Dey (2024) [8]	Deep Reinforcement Learning	Simulated Smart Grid Data	Cyber-physical attacks	91.5	92.0	93.5
Khan et al. (2024) [7]	Machine Learning (Random Forest)	Smart Grid Cyber Attack Dataset	False Data Injection, APT	87.8	86.7	89.2
Sewak et al. (2023) [6]	Deep Reinforcement Learning (PPO)	IoT Network Traffic Dataset	APT, DoS, Ransomware	92.1	91.5	92.8

6. DISCUSSION

The proposed ProAPT model, powered by DQN, offers a compelling approach for enhancing the cybersecurity of smart grids by enabling proactive and adaptive responses to Advanced Persistent Threats (APTs). The model's strong performance—achieving over 92% accuracy, precision, and recall—demonstrates its effectiveness in detecting complex attack patterns, particularly in highly dynamic IIoT environments. One of the key strengths of the ProAPT model lies in its ability to continuously learn and adapt to new threats using reinforcement signals from the environment. Unlike traditional machine learning models that rely on static rules or labeled datasets, the DRL-based approach can dynamically adjust its policies based on feedback, making it especially suitable for environments where attack vectors evolve rapidly.

In this paper, feature selection was guided both by domain knowledge and empirical importance measures derived from training the DRL model. Specifically, features such as packet size variance, connection duration, number of failed login attempts, and inbound/outbound byte ratios were selected due to their proven relevance in

identifying abnormal behaviors associated with APTs. These features reflect the temporal and statistical properties of network flows that are often manipulated during different stages of an attack, such as reconnaissance, lateral movement, or data exfiltration. To further validate their influence on DRL decision-making, we conducted a permutation-based feature importance analysis, revealing that traffic-related features had the highest impact on the agent's Q-value updates. For instance, packet size variance was frequently associated with stealthy data transfers, while connection duration helped differentiate between persistent sessions initiated by malicious actors and short-lived benign activity. By incorporating these features into the state representation, the DRL agent learned to prioritize observations that carry strong signals of attack behavior, thereby enhancing its ability to make accurate, context-aware decisions in real-time. Incorporating feature selection and emphasizing its impact on DRL decision-making helps provide a deeper understanding of how the model works and why certain features are critical for success in detecting and mitigating APTs in IIoT environments.

However, translating this success to real-world deployment scenarios presents several challenges that merit further discussion. Scalability is one such concern. While the ProAPT model performs well in controlled simulations, deploying it across large-scale, heterogeneous smart grid infrastructures may require distributed training frameworks or federated learning approaches to handle high-volume data streams without overwhelming central systems.

Another important consideration is computational efficiency and real-time responsiveness. Although DQN provides a solid balance between performance and complexity, models like Double DQN introduce architectural overhead that may hinder real-time inference in latency-sensitive applications. In this study, we observed that Double DQN, despite its theoretical advantage in mitigating Q-value overestimation, slightly underperformed compared to standard DQN. This was likely due to slower convergence in environments with strong temporal dependencies, as found in the CICAPT-IIoT dataset. Nevertheless, this does not diminish the potential of Double DQN; rather, it emphasizes the importance of careful hyperparameter tuning and task-specific architecture selection. For example, techniques such as prioritized experience replay, reward shaping, or even incorporating temporal abstraction (e.g., options frameworks or recurrent networks) may enhance the model's ability to capture long-term attack strategies while preserving inference speed. To address real-time decision-making constraints, future implementations could leverage lightweight model compression techniques (e.g., pruning, quantization) or offload computations to edge-cloud collaborative architectures. Such hybrid setups allow for scalable deployment without compromising responsiveness. Furthermore, the integration of explainability mechanisms—such as attention layers, saliency maps, or SHAP values—can significantly improve the trustworthiness of DRL decisions in operational contexts. This aligns with ongoing efforts in critical infrastructure security, where human operators require transparent and justifiable decision-making processes to support real-time incident response.

In summary, while the proposed ProAPT model demonstrates excellent potential as a next-generation defense mechanism for smart grids, addressing its implementation challenges through targeted enhancements can further solidify its applicability. The insights gained from this study also underscore the importance of balancing model sophistication with practicality, suggesting promising directions for future research in explainable, scalable, and robust DRL-based cybersecurity systems.

7. CONCLUSION

The ProAPT model showcases the promise of DRL in enhancing smart grid cybersecurity by predicting and mitigating APTs. With high performance metrics—accuracy of 92.5%, precision of 91.8%, and recall of 93.2%—the model proves its ability to detect complex attack sequences in real-time. One of the model's strengths lies in the engineering of its state space and the careful selection of relevant features, such as packet size variance, connection duration, and protocol usage. These features provide critical insights into network behavior, making the model more efficient and effective in detecting attacks. By focusing on the most important features, the model reduces computational complexity, improves accuracy, and enhances the interpretability of its decisions.

However, there are still significant challenges to overcome in deploying this model in real-world smart grid environments. The scalability of the model must be improved to accommodate larger systems with vast amounts of data, and real-time adaptability must be enhanced to respond to new attack patterns. Furthermore, the interpretability of DRL models must be addressed to ensure that cybersecurity professionals can trust and understand the model's decisions in critical infrastructure contexts.

Future work should focus on addressing these challenges by improving scalability, integrating additional data sources for enhanced predictive accuracy, and enhancing the model's interpretability. Additionally, reducing false positives will be crucial for ensuring that the system can operate without causing unnecessary disruptions. Exploring hybrid models that combine DRL with other machine learning techniques could further enhance the robustness of the ProAPT model, enabling it to better handle new and emerging threats. Finally, incorporating explainability into DRL models, especially for applications in high-stakes environments like smart grids, will be essential to ensure that automated systems can work effectively alongside human experts.

In conclusion, while the ProAPT model demonstrates great potential, ongoing research and development are necessary to refine its scalability, adaptability, and transparency, ensuring that it can provide reliable and effective protection against the evolving landscape of smart grid cybersecurity threats.

8. NOMENCLATURE & UNITS

IoT	Internet of Things
APT	Advanced Persistent Threats
IDS	Intrusion Detection Systems
DRL	Deep Reinforcement Learning
M2M	Machine-to-Machine
LSTM	Long Short Term Memory
MDP	Markov Decision Process
DQN	Deep Q-Networks
IIoT	Industrial Internet of Things

9. REFERENCES

[1] W. Wang and Z. Lu. (2013). Cyber Security in the Smart Grid: Survey and Challenges. *Computer Networks*. [Online]. *57(5)*, pp. 1344–1371. Available: https://doi.org/10.1016/j.comnet.2012.12.017

[2] M. Dehghan and E. Khosravian. (2023). Private Federated Learning for APT Detection in Internet of Drones. *Quarterly Scientific Journal of National University of Skills*. [Online]. *20(3)*, pp. 465–484. Available: https://karafan.tvu.ac.ir/article_179732.html?lang=en

[3] Gunduz, M. Z., & Das, R. (2020). Cyber-security on smart grid: Threats and potential solutions. *Computer networks*. [Online]. *169*, p. 107094. Available: https://doi.org/10.1016/j.comnet.2019.107094

[4] Z. Ding, Y. Huang, H. Yuan, and H. Dong. (2020). Introduction to Reinforcement Learning. *Deep Reinforcement Learning: Fundamentals, Research and Applications, Singapore: Springer Singapore*. [Online]. pp. 47–123. Available: https://doi.org/10.1007/978-981-15-4095-0_2

[5] N. Abdi, A. Albaseer, and M. Abdallah. (2024). The Role of Deep Learning in Advancing Proactive Cybersecurity Measures for Smart Grid Networks: A Survey. *IEEE Internet of Things Journal*. [Online]. *11(9)*, pp. 16398–16421. Available: https://doi.org/10.1109/JIOT.2024.3354045

[6] M. Sewak, S. K. Sahay, and H. Rathore. (2023). Deep Reinforcement Learning in the Advanced Cybersecurity Threat Detection and Protection. *Information Systems Frontiers*. [Online]. *25(2)*, pp. 589–611. Available: https://doi.org/10.1007/s10796-022-10333-x

[7] Khan, M. A., Saleh, A. M., Waseem, M., & István, V. (2024, Sep.). Smart Grid Cyber Attacks: Overview, Threats, and Countermeasures. In 2024 22nd International Conference on Intelligent Systems Applications to Power Systems (ISAP). [Online]. pp. 1-5. Available: https://doi.org/10.1109/ISAP63260.2024.10744349

[8] S. Maiti, S. Adhikary, S. Dey, and A. R. Hota. (2024). Learning-Enabled Adaptive Voltage Protection Against Load Alteration Attacks On Smart Grids. *arXiv preprint*. [Online]. Available: https://doi.org/10.48550/arXiv.2411.15229

[9] Y. Yang, T. Littler, S. Sezer, K. McLaughlin, and H. F. Wang. (2011, Dec.). Impact of Cyber-Security Issues on Smart Grid. In International Conference and Exhibition on Innovative Smart Grid Technologies, pp. 1–7. [Online]. Available: https://doi.org/10.1109/ISGTEurope.2011.6162722

[10] Khosravian, E. and Dehghan, M. (2025). Cyber Risk Prediction for UAVs in Space-Related Missions Using Deep Reinforcement Learning. *Journal of Space Science and Technology*. [Online]. *18*, pp. 1-15. Available: https://doi.org/10.22034/jsst.2025.1527

[11] M. Dehghan , B. Sadeghiyan , E. Khosravian , A. Sedighi Moghadam and F. Nooshi. (2025). ProAPT: Projection of APTs with Deep Reinforcement Learning. The ISC International Journal of Information Security. 17 (1). pp. 25-41, doi: 10.22042/isecure.2024.428569.1052

[12] Ghiasvand, E., Ray, S., Iqbal, S., Dadkhah, S., & Ghorbani, A. A. (2024). CICAPT-IIOT: A provenance-based APT attack dataset for IIoT environment. *arXiv preprint*. [Online]. Available: https://doi.org/10.48550/arXiv.2407.11278

[13] Shees, A., Tariq, M., & Sarwat, A. I. (2024). Cybersecurity in Smart Grids: Detecting False Data Injection Attacks Utilizing Supervised Machine Learning Techniques. *Energies*. [Online]. 17(23), p. 5870. Available: https://doi.org/10.3390/en17235870

[14] S. Maiti and S. Dey. (2024). Smart Grid Security: A Verified Deep Reinforcement Learning Framework to Counter Cyber-Physical Attacks. *arXiv preprint*. [Online]. Available: https://doi.org/10.48550/arXiv.2409.15757

[15] H. Biswas. (2024, Sep.). Malware Trend in Smart Grid Cyber Security. *IEEE Region 10 Symposium (TENSYMP)*. [Online]. pp. 1–5. Available: https://doi.org/10.1109/TENSYMP61132.2024.10752141

[16] Paul, B., Sarker, A., Abhi, S. H., Das, S. K., Ali, M. F., Islam, M. M., ... & Saqib, N. (2024). Potential smart grid vulnerabilities to cyber attacks: Current threats and existing mitigation strategies. *Heliyon*. [Online]. *10*(19). Available: https://doi.org/10.1016/j.heliyon.2024.e37980

[17] N. Sahani, R. Zhu, J. H. Cho, and C. C. Liu. (2023). Machine Learning-Based Intrusion Detection for Smart Grid Computing: A Survey. *ACM Transactions on Cyber-Physical Systems*. [Online]. *7(2)*, pp. 1–31. Available: https://doi.org/10.1145/3578366

[18] H. Shadabfar, M. Dehghan, and B. Sadeghiyan. (2024). DSRL-APT-2023: A New Synthetic Dataset for Advanced Persistent Threats. In *21st International ISC Conference on Information Security and Cryptology (ISCISC 2024)*. [Online]. Available: https://doi.org/10.22042/isecure.2025.214212

[19] D. Kushner. (2013). The Real Story of Stuxnet. *IEEE Spectrum*. [Online]. *50(3)*, pp. 48–53.

[20] Khan, R., Maynard, P., McLaughlin, K., Laverty, D., & Sezer, S. (2016, August). Threat analysis of blackenergy malware for synchrophasor based real-time control and monitoring in smart grid. In 4th International Symposium for ICS & SCADA Cyber Security Research. [Online]. pp. 53-63.

[21] A. S. AL-Aamri, R. Abdulghafor, S. Turaev, I. Al-Shaikhli, A. Zeki, and S. Talib. (2023). Machine Learning for APT Detection. *Sustainability*. [Online].

15(18), p. 13820. Available: https://doi.org/10.3390/su151813820

[22] Nguyen, T. T., & Reddi, V. J. (2021). Deep reinforcement learning for cyber security. *IEEE Transactions on Neural Networks and Learning Systems*. [Online]. *34*(8), 3779-3795. Available: https://doi.org/10.1109/TNNLS.2021.3121870

[23] E. M. S. P. Veith, A. Wellßow, and M. Uslar. (2023). Learning new attack vectors from misuse cases with deep reinforcement learning. *Frontiers in Energy Research*. [Online]. *11*, p. 1138446. Available: https://doi.org/10.3389/fenrg.2023.1138446

[24] A. Sinha, R. Vyas, F. Alasali, W. Holderbaum, and O. P. Vyas. (2025). A deep reinforcement learning-based approach for cyber resilient demand response optimization. *Frontiers in Energy Research*. [Online]. *12*, 1494164. Available: https://doi.org/10.3389/fenrg.2024.1494164

[25] Y. Li, W. Ma, Y. Li, S. Li, Z. Chen, and M. Shahidehpour. (2025). Enhancing Cyber-Resilience in Integrated Energy System Scheduling with Demand Response Using Deep Reinforcement Learning. *Applied Energy*. [Online]. *379*, p. 124831. Available: https://doi.org/10.1016/j.apenergy.2024.124831

[26] Q. Sun, G. Lian, Z. Cao, X. Zeng, Z. Lv, L. Liu, ... and T. X. Zheng. (2023, Sep.). Deep Reinforcement Learning Based Secure Communication and Computing Resource Allocation for Grid Cyber-Physical System. In Proceedings of the 2nd International Conference on Internet of Things, Communication and Intelligent Technology, pp. 274–283. Singapore: Springer Nature Singapore. [Online]. Available: https://doi.org/10.1007/978-981-97-2757-5_29

[27] M. Dehghan and B. Sadeghiyan. (2018, May). An Efficient Secure Generalized Comparison Protocol. In *Electrical Engineering (ICEE), Iranian Conference on*, pp. 1487–1492. [Online]. Available: https://doi.org/10.1109/ICEE.2018.8472437

[28] Carvalho, D. V., Pereira, E. M., & Cardoso, J. S. (2019). Machine learning interpretability: A survey on methods and metrics. *Electronics*. [Online]. *8(8)*, p. 832. Available: https://doi.org/10.3390/electronics8080832

[29] M. Dehghan and B. Sadeghiyan. (2020, Oct.). Secure Multi-Party Sorting Protocol Based on Distributed Oblivious Transfer. In 10th International Conference on Computer and Knowledge Engineering (ICCKE), pp. 011–017. [Online]. Available: https://doi.org/10.1109/ICCKE50421.2020.9303630

[30] Dehghan, M. , Mahdi Zadeh, A. and Sadeghian, B. (2024). *A* Model to Measure Effectiveness in Cyber Security Situational Awareness. *Computer and Knowledge Engineering*. [Online]. 7(1), pp. 17-26. Available: https://doi.org/10.22067/cke.2024.83723.1101

[31] M. Dehghan and E. Khosravian. (2024). A Review of Cognitive UAVs: AI-Driven Situation Awareness for Enhanced Operations. *AI and Tech in Behavioral and Social Sciences*. [Online]. *2(4)*, pp. 54–65. [Online]. Available: https://doi.org/10.61838/kman.aitech.2.4.6

[32] M. Saarela and S. Jauhiainen. (2021). Comparison of Feature Importance Measures as Explanations for Classification Models. SN Applied Sciences. [Online]. 3(2), p. 272. Available: https://doi.org/10.1007/s42452-021-04148-9

Journal of Computer and Knowledge Engineering, Vol. 8, No. 2, 2025. (41-54)

Ferdowsi University of Mashhad

Journal of Computer and Knowledge Engineering

https://cke.um.ac.ir

Information and Communication Technology Association of Iran

Structure Optimization in Deep Neural Networks with Synaptic Pruning Based on Connection Appraisal*

Research Article

Aghil Ahmadi[1], Reza Mahboobi Esfanjani[2]

10.22067/cke.2025.88039.1113

Abstract *Deep neural networks typically require predefined architectures, which can lead to overfitting, underfitting, high computational costs, and storage overhead. Dynamic structure optimization through pruning can reduce network redundancy but often results in performance degradation. In this study, we propose a novel pruning method inspired by biological synaptic pruning that adaptively optimizes deep neural network structures. The proposed method continuously monitors the contribution of each connection during training using a dynamic efficiency criterion that evaluates the relative importance of each connection within its layer. Connections are not removed immediately; instead, only those consistently falling below a predefined threshold are pruned, ensuring stability and robustness. Simulation validation is conducted on an industrial distillation column dataset under noisy conditions and the MNIST benchmark dataset. The results demonstrate improved accuracy, enhanced generalization, and faster learning, with an average pruning rate of 53%. Compared to conventional and state-of-the-art pruning techniques, our method achieves superior performance in terms of compression rate and accuracy while effectively mitigating overfitting.*

Key Words Deep Neural Networks, Synaptic Pruning, Distillation Column, Connection Evaluation.

1. INTRODUCTION

In the fields of data science and artificial intelligence, machine learning has experienced tremendous growth. Among its various tools, artificial neural networks (ANNs) have become some of the most reliable and widely used methods, owing to their parallel distributed architecture, learning capability, and generalization potential [1]. These features enable neural networks to effectively handle complex tasks such as automatic control, system identification, and pattern recognition.

A neural network's structure (comprising the number of hidden layers and associated weights) plays a crucial role in determining its overall performance. Both excessively small and overly large networks pose challenges: small networks lack sufficient capacity to model complex relationships, making them difficult to train, while large networks suffer from overfitting, reduced generalization, and increased computational burden [2], [3], [4]. Achieving an optimal network size is thus vital for creating models that are not only accurate but also efficient and interpretable. The recent success of deep neural networks (DNNs) in various machine learning applications has further highlighted this trade-off. Despite their superior performance, DNNs typically demand substantial memory and processing power, making them difficult to deploy in environments with limited computational resources, such as mobile devices and embedded systems [5], [6]. Consequently, methods to reduce the complexity of these networks without sacrificing accuracy have become essential. One widely adopted solution is neural network pruning, which systematically removes unnecessary parameters from a trained network to simplify its structure. Pruning can effectively reduce computational and storage overhead while maintaining acceptable levels of accuracy.

* Manuscript received 2024 May 12, Revised 2025 May 2 Accepted 2025 June 28.

[1] M.Sc. Department of Electrical and Computer Engineering, Sahand University of Technology, Tabriz, Iran.
Email: ag_ahmadi@sut.ac.ir

[2] Corresponding Author. Professor,Department of Electrical and Computer Engineering, Sahand University of Technology, Tabriz, Iran. **Email:** mahboobi@sut.ac.ir

Although pruning has been explored since the late 1980s [7], its relevance has resurfaced with the growing depth and complexity of modern networks. A fundamental challenge in pruning is identifying which connections are suitable candidates for removal. Traditional methods often rely on the magnitude of the connection weights, assuming that smaller weights contribute less to the network's output and thus can be safely pruned. However, both theoretical studies and empirical evidence have shown that this assumption can be misleading [2], [3]. Important connections may exhibit small weight magnitudes due to specific data distributions or network dynamics. Consequently, relying solely on weight magnitude as a pruning criterion risks discarding valuable connections and potentially degrading network performance.

Recent research has emphasized the need for more robust evaluation criteria that go beyond simple weight magnitude. However, many existing methods still assess connection importance in a static, single-phase manner without continuously monitoring their contribution during the training process. Furthermore, most of these approaches focus primarily on optimization and regularization objectives, lacking a biologically plausible foundation [4]. In contrast, the human brain offers a compelling model for effective pruning. During development, the brain undergoes synaptic pruning, a process where redundant or weak synaptic connections are gradually eliminated based on their activity levels [8]. This activity-dependent mechanism strengthens frequently used synapses while removing those that are rarely activated, leading to a more efficient and specialized network [9], [10], [11]. Incorporating such biologically inspired strategies into artificial neural network pruning can potentially enhance both effectiveness and robustness. In this paper, we propose a novel method for optimizing the structure of deep neural networks by integrating brain-inspired synaptic pruning mechanisms with connection evaluation based on network error contribution. Unlike traditional methods that rely on weight magnitude, our approach dynamically monitors the actual influence of each connection on the network's performance. Connections with persistently weak contributions are gradually eliminated, mirroring the brain's "use it or lose it" principle. This strategy not only reduces the risk of removing valuable connections but also improves the network's ability to handle noisy and uncertain data. The remainder of this paper is organized as follows: Section 2 reviews related works in neural network pruning and structure optimization; Section 3 presents the proposed pruning method; Section 4 provides comparative results and discussion; and finally, Section 5 concludes the paper.

2. RELATED WORKS

The primary distinction between shallow and deep neural networks lies in the number of hidden layers. Shallow networks typically consist of a single hidden layer, whereas deep networks comprise multiple hidden layers (at least three), enabling hierarchical feature extraction and improved representation of complex data. This hierarchical structure enhances robustness in managing uncertainties and allows deep networks to model more precise functions, making them superior in applications requiring complex feature learning, such as industrial process modeling and control.

One of the earliest solutions for reducing the computational complexity of neural networks (NNs) is knowledge distillation, in which a smaller model is trained to mimic the behavior of a larger, well-trained model [12]. Despite its effectiveness, this approach requires predefined architectures for student networks, which limits flexibility.

Another extensively studied method is network pruning, where neurons or connections with minimal contribution are systematically removed. Traditional pruning techniques often rely on thresholding weight magnitudes, assuming that smaller weights are less significant [13], [14]. However, this approach has been questioned, as critical connections might occasionally have small weight magnitudes depending on the data and network dynamics [15].

To address these limitations, more advanced pruning criteria have been introduced. For instance, Molchanov et al. [16] proposed utilizing feature map statistics and mutual information to evaluate the relevance of connections. Other researchers have adopted Taylor series expansions for sensitivity analysis, such as the first-order approach by Molchanov et al. [16] and second-order methods by LeCun et al. [17] and Hassibi and Stork [18], using Hessian approximations for more accurate significance estimation.

Beyond individual weights, filter-level pruning methods have also emerged. He et al. [19] proposed a geometric median-based method for removing redundant filters. Yu et al. [20] introduced the Neuron Importance Score Propagation (NISP) technique, propagating importance values backward through the network layers. Li et al. [21] focused on pruning filters with lower weights, and He et al. [22] introduced soft filter pruning, allowing pruned filters to recover through retraining. While these methods improve computational efficiency, they often lack biological plausibility, focusing on mathematical heuristics rather than biologically inspired mechanisms. Furthermore, many existing approaches perform one-time static evaluations without continuously monitoring the dynamic role of connections during training. Regularization methods such as dropout [23] and dropconnect [24] have been effective in preventing overfitting by randomly deactivating neurons or weights during training. However, they do not reduce network complexity at inference time, as all connections are reactivated. Similarly, techniques like meProp [25] sparsify gradients during backpropagation to accelerate

training but do not alter the network's structure.

Another prominent line of research involves evolutionary algorithms, which simultaneously optimize network topology and weights. Evolutionary strategies employ fitness functions that consider accuracy and network complexity [26], [27], [28]. Genetic algorithms, in particular, have been used to prune networks and discover efficient topologies [29], [30], [31], [32]. Despite their adaptability, these methods are computationally intensive and suffer from convergence uncertainties due to the vast search space.

Recently, several pruning methods have been proposed to enhance the efficiency of deep neural networks without significantly compromising performance. In the Lottery Ticket Hypothesis (LTH) method, the concept of "winning tickets" was used to train small subnetworks that can match the performance of the original network if initialized properly [33]. SNIP presents a pre-training pruning strategy based on connection sensitivity to loss, allowing efficient identification of crucial weights before training [34]. GraSP further improves pruning by preserving the gradient flow essential for learning [35]. Movement Pruning is a dynamic pruning method applied during fine-tuning, focusing on the directional movement of weights to identify unimportant connections [36]. Additionally, Global Magnitude Pruning selects the weakest weights across the entire network rather than layer by layer, achieving a better balance between sparsity and accuracy [37]. Despite their success, most of these methods rely heavily on initial weight magnitudes or static criteria, whereas our proposed method continuously monitors the dynamic contribution of each connection to the network's error during training, inspired by biological synaptic pruning mechanisms.

In summary, while significant progress has been made in neural network pruning with the advent of the Lottery Ticket Hypothesis (LTH), SNIP, GraSP, Movement Pruning, and Global Magnitude Pruning, these approaches still primarily rely on static evaluations or single-shot sensitivity analyses. They often assess connection importance based on initial weight magnitudes, gradient sensitivity, or weight movement trends, with limited adaptation during the training process. Moreover, most SOTA methods lack a biologically inspired mechanism to guide pruning decisions dynamically. These gaps highlight the necessity for pruning strategies that can adapt to the evolving structure and error dynamics of the network. Our proposed method addresses these limitations by continuously monitoring the real-time contribution of each connection to the overall network error and gradually pruning redundant connections, inspired by the synaptic pruning process observed in biological neural systems. This dynamic and brain-inspired approach ensures more robust pruning decisions and greater resilience to noisy and uncertain data, pushing beyond the capabilities of existing SOTA techniques.

3. PROPOSED PRUNING METHOD

We present an innovative Brain-Inspired Connection Evaluation Pruning technique in this section. In the first stage of the proposed algorithm, the real value of each connection in the network is determined, which essentially reflects the importance and contribution of that connection to the overall network performance. In this context, the "real value" is assessed based on an error-driven criterion, where the impact of omitting each connection on the network's output error is evaluated. This allows for a more accurate measurement of each connection's significance beyond simple weight magnitudes.

This evaluation is based on neglecting each connection and computing the error that results from its removal. To measure the true value of the neurons, the current output of the network must be brought closer to the ideal values. In other words, connections that lead to a deviation of the output from the ideal values increase errors. We arrange the connections according to the value of training errors produced when they are eliminated. In the process of pruning, our goal is to make the network lighter and smaller, but we must note that the accuracy of the network should not decrease too much. Therefore, pruning candidates include a subset of connections that have produced the minimum value of errors. We will delete connections inspired by the pruning process in the human brain as follows: brain pruning involves making stronger connections with a higher frequency of use and weaker connections with a lower frequency of use [38]. A connection will be deleted if, over the course of several steps, its strength falls below a predetermined threshold. Namely, if the weak score of a connection persists, it will be eliminated. Fig. 1 depicts the process of synaptic pruning. It is evident that we need to specify two crucial parameters. The first parameter is the threshold, which indicates which connections may need pruning. The second parameter is the warning time, which indicates how long the related connection will remain active before being deleted. However, here the criterion is training error instead of connection weights.

Use it or lose it: neuroscientists refer to the decrease in spine density as "synaptic pruning." Through this process, weaker structures are eliminated, reallocating resources to the surviving ones so they can become stronger and more stable. As it became abundantly evident that synaptic activity directs appropriate pruning, scientists focused on identifying the cellular processes that might control the remodeling [38].

We determine a threshold value for the acceptable error in order to guide the pruning process. This threshold serves as a benchmark to evaluate the significance of each connection within the network. During each iteration, we closely monitor the connections whose removal results in

minimal increases in error compared to the previous step. These connections, having demonstrated a consistently low impact on overall network performance, are considered potential candidates for removal. To ensure a cautious and reliable pruning process, we introduce a control mechanism known as the "warning number." This parameter defines the required consistency of a connection's low contribution across multiple evaluations. Specifically, connections that remain below the error threshold for a certain number of consecutive iterations (defined by the warning number) are identified as weak contributors and selected for pruning. This progressive evaluation prevents the premature removal of connections that might exhibit temporary fluctuations in importance due to network dynamics. This method allows for a gradual and robust reduction in network complexity, as only the connections with persistently negligible impact are pruned. By continuously reassessing the error contribution of each connection, the proposed approach mimics biological pruning mechanisms, ensuring that only truly redundant connections are eliminated. The procedure of the proposed pruning technique is illustrated in Fig. 2, which visually represents the step-by-step process, including error evaluation, candidate selection, application of the warning number criterion, and final pruning decisions.

The pruning pseudo-code is presented in detail in Table 1. This combined pruning method is presented to address the disadvantages of existing pruning methods as mentioned in the previous sections: relying only on the weighted domain is not sufficient, and there is a high probability that some very important network connections are omitted. We addressed this weakness by sorting the connections, and after finding the connections susceptible to deletion, the removal is not done in one step by decreasing the value once. We successively caution the pruning candidates and prune them based on these warnings.

In summary, the evaluation of all network connections is carried out based on their contribution to the overall network error. Specifically, we determine the error introduced by individually removing each connection and then rank the connections according to the magnitude of these errors. Connections associated with the least error increases are considered for removal, guided by a pruning rate defined by the designer. Consequently, our pruning strategy incorporates two key elements: evaluating connections based on training error and tracking their iterative weak scores. Ultimately, this process yields a pruned network that significantly outperforms the original configuration. The motivation behind the proposed pruning strategy stems from the limitations observed in existing methods. Most conventional pruning techniques rely heavily on static evaluations, primarily based on weight magnitude or sensitivity analyses performed either before or after training. Such static approaches often fail to capture the dynamic behavior and real-time importance of connections throughout the learning process, leading to the risk of pruning significant but low-magnitude connections and potentially degrading network performance.

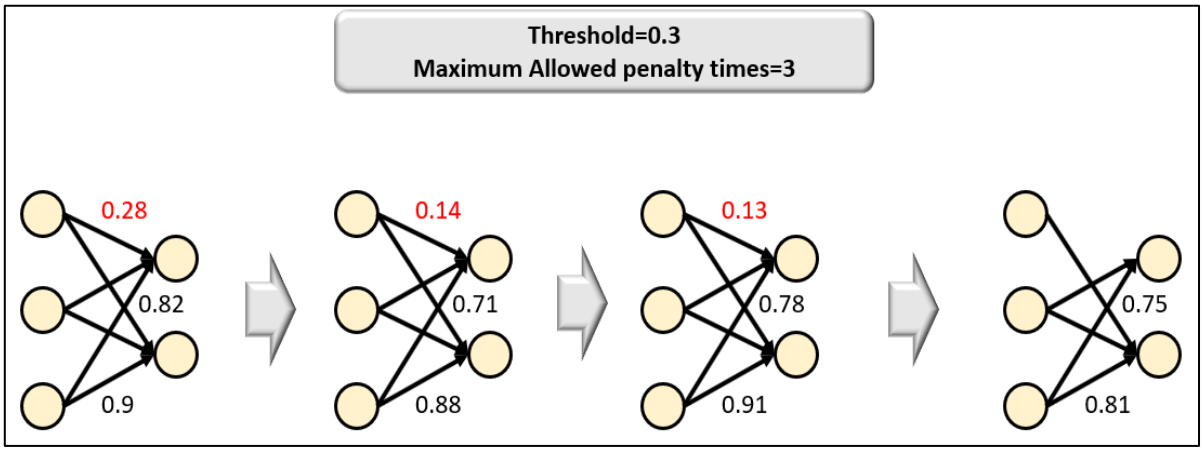


Fig. 1. Process of synaptic pruning

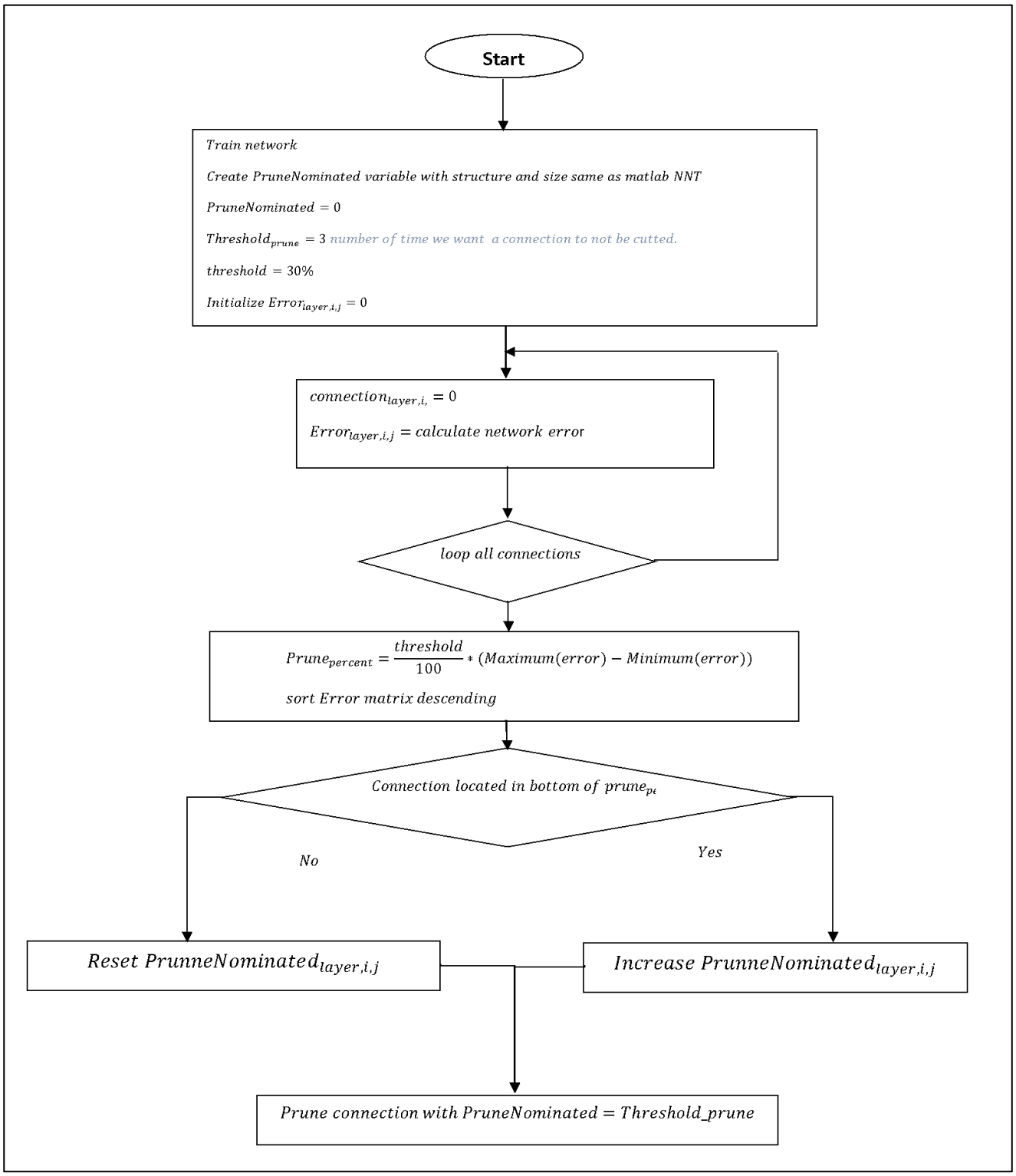


Fig. 2. Flowchart of the proposed pruning

TABLE1
Pruning pseudo code

Train network

Create PruneNominated variable with structure and size same as matlab NNT

$\text{PruneNominated} = 0$

$\text{Threshold}_{\text{prune}} = 3$ number of time we want a connection to not be cutted.

$\text{threshold} = 30\%$

Initialize $\text{Error}_{\text{layer,i,j}} = 0$

loop steps each second

loop connections

$\text{connection}_{\text{layer,i,}} = 0$

$\text{Error}_{\text{layer,i,j}} = \text{calculate network error}$

end loop

$\text{Prune}_{\text{percent}} = \frac{\text{threshold}}{100} * (\text{Maximum}(\text{error}) - \text{Minimum}(\text{error}))$

sort Error matrix descending

for error counter $= 1$ to $\text{count}_{\text{connections}} - \text{Prune}_{\text{percent}} * \text{count}_{\text{connections}}$

$\text{PruneNominated}_{\text{connectiont}_{\text{error counter}}} = 0$

end for

for error counter $= \text{count}_{\text{connections}} - \text{Prune}_{\text{percent}} * \text{count}_{\text{connections}} + 1$ to end

$\text{PruneNominated}_{\text{connectiont}_{\text{error counter}}} + +$

end for

loop connections

if $\text{PruneNominated}_{\text{connection}} = \text{Threshold}_{\text{prune}}$

Prune this connection

end if

end loop

end loop

In contrast, the human brain undergoes synaptic pruning based on continuous monitoring of synaptic activity, gradually eliminating weak and unused connections while reinforcing the strong ones. Inspired by this biological process, our approach integrates a dynamic evaluation criterion that monitors the real-time contribution of each connection to the overall training error. By focusing on the impact of each connection on network performance rather than solely on its weight magnitude, we ensure that only truly redundant connections are pruned.

Moreover, the introduction of a "warning number"—requiring multiple consecutive evaluations before pruning— prevents the premature removal of connections due to temporary fluctuations, thus enhancing the robustness of the pruning process. This feature becomes particularly crucial in noisy or uncertain environments, such as industrial process modeling, where data variability can affect the stability of traditional pruning methods.

Therefore, the proposed method not only addresses the shortcomings of static and heuristic-driven pruning approaches but also offers a biologically plausible, adaptive, and noise-resilient solution for optimizing deep neural network architectures. These attributes make it a highly appropriate choice for complex, real-world applications.

4. COMPARISON RESULTS AND DISCUSSION

In this section, we apply the suggested method to a neural network model of a refinery process's distillation tower in order to assess its efficacy. The objective is to investigate how, in the case of ideal and noisy data, the proposed algorithm can enhance identification accuracy and convergence speed.

The distillation tower, which is a multi-input, multi-output (MIMO) nonlinear system, is a general and inseparable part of a refinery. A distillation column is a device for separating components of a solution. In fact, in the distillation tower, the components of a solution are separated based on their volatility and boiling point differences. Industrial distillation towers are widely used in various process industries, but one of their main uses is crude oil refinement. In the oil industry, different hydrocarbons are separated based on their volatility by the distillation method. The ethane-ethylene distillation column is one of the most widely used towers. Due to its significance, high-purity ethylene production is required. Our data belongs to an ethane-ethylene distillation column identification experiment. There are four series in the data [39]:

U_dest, Y_dest: without noise (ideal series)
U_dest_n10, Y_dest_n10: 10 percent additive white noise
U_dest_n20, Y_dest_n20: 20 percent additive white noise
U_dest_n30, Y_dest_n30: 30 percent additive white noise

There are 90 samples for neural network training. The following describes the inputs and outputs:

Inputs:

1) The proportion between feed flow and reboiler duty
2) The relationship between feed flow and reflux rate
3) Proportion between the feed flow and the distillate
4) Composition of input ethane
5) Top pressure

Outputs:

1) Top ethane composition
2) Bottom ethylene composition
3) Top-bottom differential pressure.

Therefore, we use a deep network with 5 inputs and 3 outputs and also 90 connections (Fig. 3). We can leverage the capabilities of the deep network, provided that we first have correct weight training and, secondly, to increase the speed of the network and prevent overfitting, we find the best possible structure for the network through our structural optimization scheme.

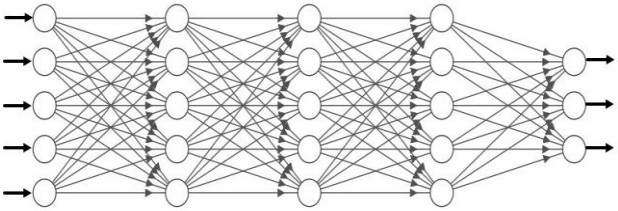


Fig. 3. Applied Deep Neural Network

First, we train the network with the data we have. Fig. 4 shows how the network performance changes (performance function value) each time the network is trained. It includes three curves with different colors for training, validation, and test data. The value of the performance function on the data in each category is displayed in each plot. The horizontal axis label indicates the number of times (epochs) the network has been trained. Also, the title of this graph shows that the best performance of the network (on training and validation data) was achieved in the second epoch, along with the value of the performance function at this point. This optimal point is also marked by two crossed dotted lines whose intersection is at the optimal point, and a green circle is drawn around this point. Furthermore, the regression charts for the training, validation, and test data are given in Fig. 5.

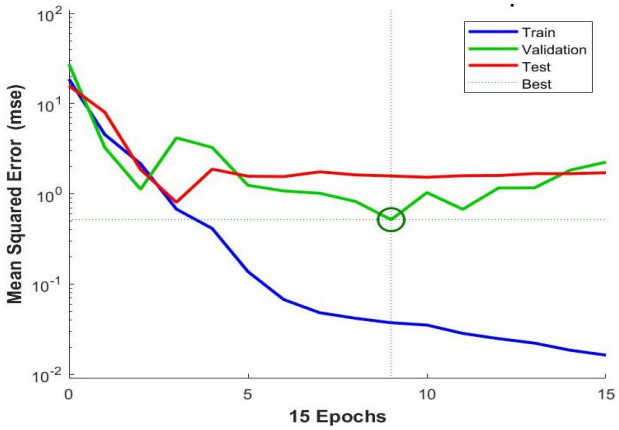


Fig. 4. Performance of the deep neural network

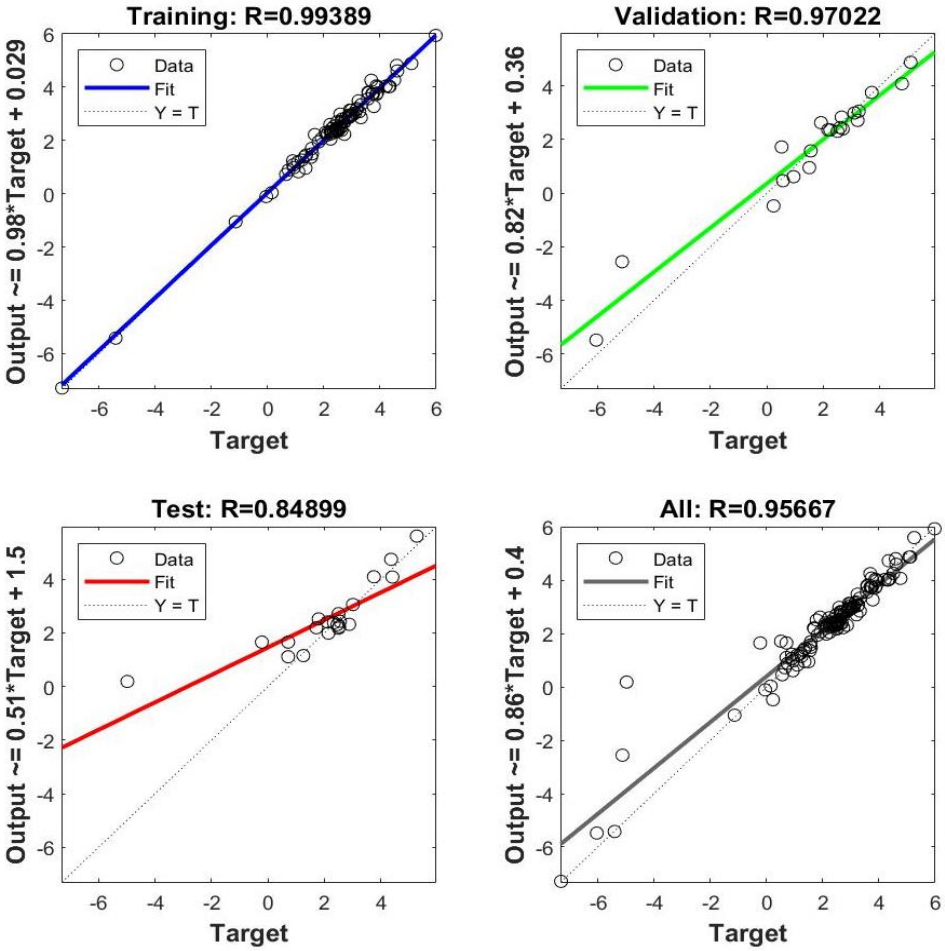


Fig. 5. Regression for the training, validation and test data

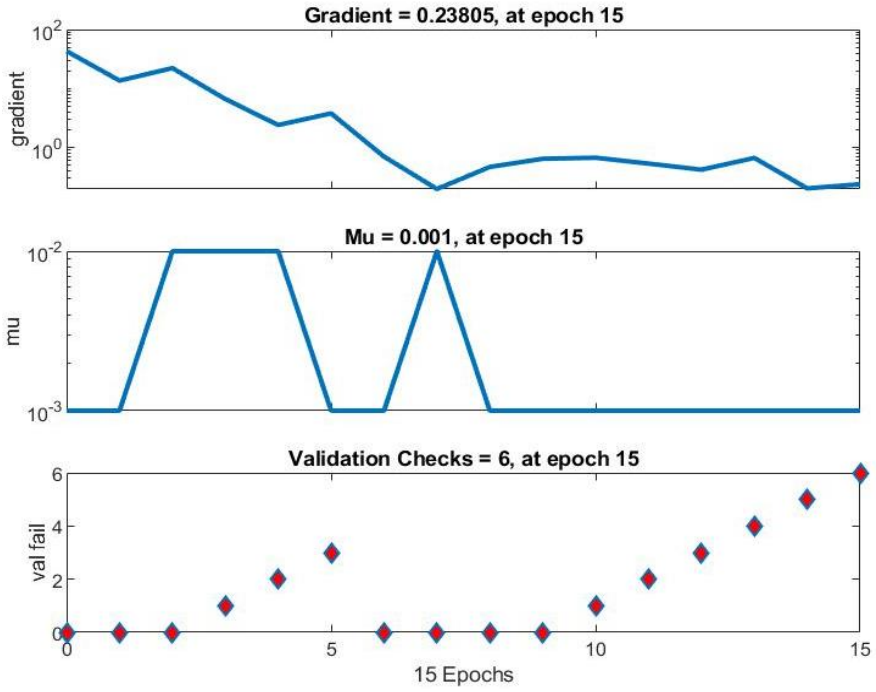


Fig. 6. Training state

In the training state visualization shown in Fig. 6, more information from the training is displayed; for example, the "val fail" graph shows in which epoch the evaluation of the validation data was rejected. This graph shows the cumulative number of failed evaluations. Training stops whenever the network fails six consecutive evaluations.

Comparing different pruning techniques to assess how far the field has come in recent years is a challenging task. Nonetheless, two significant metrics are typically employed and presented here. The compression ratio is defined as the new size divided by the original size. The theoretical speedup is defined as the ratio of the initial number of multiply-adds to the new number. The performance function is the function on which the performance of the network is measured. In this problem, our performance function is MSE (mean square error). In Table 2, comparative data are shown for different scenarios. We study networks with different topologies (shallow and deep) and also compare our approach to the dropout method [24], which is a powerful technique to prevent overfitting under similar circumstances.

As seen, we accelerated network performance and training by utilizing an inventive pruning technique. It is simple to expand the suggested pruning method to other intelligent process industries. Noisy data, which is commonly encountered in real-world industrial settings, is one of the most significant issues in measurement and control. This work aims to investigate whether the proposed algorithm can enhance the speed of convergence and identification accuracy even in cases where a large number of connections are ignored and, more crucially, the data is noisy.

The results of the deep network pruned using the proposed approach, presented in Fig. 7, are compared with those of the shallow network when dealing with data that is noise-free, with 10%, 20%, and 30% noise. It is evident that the proposed structure performs noticeably better, particularly when handling noisy data.

Concisely, a deep network pruned with the proposed method is used to model the distillation tower, and its efficiency was demonstrated compared to the shallow network. Additionally, we compared (Table 3) the RMSE criterion between the proposed model and three other structures in order to compare it with other neural network-based models. The mentioned structures are: nonlinear auto-regressive with exogenous inputs (NARX)-based ANFIS and NARX structure-based neural networks (using both the Levenberg–Marquardt and the Steepest Descent algorithms) [40]. The comparison of errors amply demonstrates the superiority of the proposed method over alternative structures.

TABLE 2
Comparative results

NN type	**Shallow NN (1 hidden layer)**			**NN (2 hidden layers)**			**Deep NN (3 hidden layers)**		
Parameters	Initial	ropout	PROPOSED	Initial	Dropout	PROPOSED	Initial	Dropout	PROPOSED
Accuracy (%)	76.62	77.10	77.94	81.35	82.70	84.73	82.63	83.87	85.89
Net. Compression %)	-	47	47.26	-	53	53.16	-	58	58.71
Execution Time (ms)	15	17.4	13.5	16	18.7	14.45	17.5	19.2	15

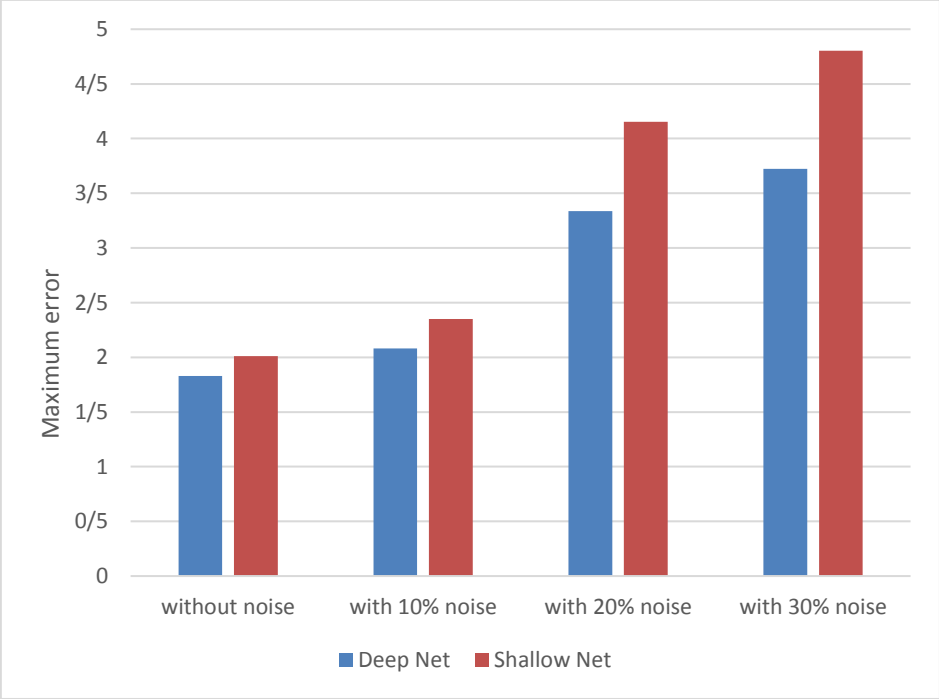


Fig. 7. Deep and Shallow networks comparison in noisy data management

TABLE 3
RMSE for neural networks models, ANFIS and the proposed

Outputs	**Steepest Descent**	**Levenberg Marquardt**	**ANFIS**	**PROPOSED**
Top Composition	0.639	0.2090	0.0421	0.0233
Bottom Composition	1.3127	0.4913	0.031	0.024
Pressure Difference	1.0053	0.2480	0.0189	0.0117

4.1. Generalization Capability

Although this study focused on the distillation column dataset, the underlying principles of the proposed pruning method are generalizable to other complex, nonlinear systems. The dynamic evaluation of connection contributions ensures that the method adapts to diverse data patterns, making it applicable to various domains where overfitting and redundancy are significant concerns. The progressive, biologically inspired pruning strategy further enhances the model's ability to handle unseen data, supporting its potential use in broader industrial and scientific applications.

4.2. Run Time Complexity Analysis

From a computational perspective, the proposed method introduces additional overhead during training due to continuous connection evaluation. However, this overhead is strategically balanced by the significant reduction in network size, which directly impacts inference speed and computational resource requirements. The results in Table 2. highlight that despite the added complexity in the training phase, the overall execution time is reduced post-pruning. This trade-off is particularly beneficial in real-time applications where inference speed is critical. Additionally, the pruning process does not require retraining from scratch, which further mitigates computational costs. By focusing on preserving high-contribution connections, the method ensures efficiency without compromising accuracy, positioning it as a practical solution for resource-constrained environments.

4.3. Comparative Analysis of Pruning Methods

To provide a broader and more comprehensive perspective, we compared our proposed pruning method with several state-of-the-art (SOTA) approaches in the field. These include the Lottery Ticket Hypothesis (LTH), SNIP, GraSP, Movement Pruning, and Global Magnitude Pruning.

The comparison focuses on key characteristics such as the use of dynamic monitoring, biological inspiration, timing of pruning during the learning process, and robustness to noisy data.

As seen in Table 4, most of the SOTA methods focus on static or pre-training evaluations and are not inspired by biological processes. Furthermore, they generally lack robustness when dealing with noisy data, which is common in real-world industrial applications. In contrast, our proposed method incorporates dynamic monitoring of connection contributions throughout training, guided by brain-inspired synaptic pruning principles. This dynamic evaluation not only enables more precise pruning decisions but also enhances the model's ability to handle noisy datasets, as demonstrated by the experimental results. We have included a quantitative comparison between the proposed method and a conventional pruning method (Global Magnitude Pruning) and a recent state-of-the-art method, SNIP. Comparisons are made on both the industrial distillation column dataset and the MNIST benchmark dataset. The results clearly indicate that the proposed method consistently achieves higher accuracy and compression rates across both datasets. This highlights the method's potential for broader application in domains where data quality and model efficiency are critical.

The results in Table 5 indicate that our proposed method consistently outperforms both traditional and recent state-of-the-art pruning techniques in terms of accuracy and compression rate across different datasets.

TABLE 4
Comparison of pruning methods based on key characteristics

Method	**Dynamic Monitoring**	**Biologically Inspired**	**Pretraining/Post-training**	**NoisyData Robustness**
Lottery Ticket Hypothesis (LTH)	×	×	Post-training	×
SNIP	×	×	Pre-training	×
GraSP	×	×	Pre-training	×
Movement Pruning	✓	×	During fine-tuning	×
Global Magnitude Pruning	×	×	Post-training	×
Proposed Method	✓	✓	During training	✓

TABLE 5
Quantitative evaluation of the proposed method versus recent approaches

Dataset	Method	Accuracy (%)	Compression Rate (%)
Distillation Column	Global Magnitude	83.2	50%
Distillation Column	Proposed Method	85.9	58.7%
MNIST	SNIP	98.2	40%
MNIST	Proposed Method	98.5	52%

4.4. Generalization and Overfitting Control

In addition to improving model efficiency, pruning methods play a critical role in enhancing generalization by reducing network complexity. The proposed brain-inspired dynamic pruning approach continuously monitors and removes redundant connections during training, leading to a more compact network structure with fewer parameters. This reduction in the model's capacity limits its ability to overfit the training data and facilitates better generalization to unseen samples. The results reported in Table 5 further support this claim, showing minimal gaps between training and testing performance across different datasets, including the industrial distillation column and the MNIST benchmark. Such consistency in performance demonstrates that the proposed pruning strategy effectively mitigates overfitting and improves the network's generalizability, even under noisy and complex conditions.

4.5. Limitations and Future Work

While the current study provides comprehensive validation on the distillation column dataset, future work will focus on applying the proposed method to other datasets to further validate its generalizability. Nevertheless, the algorithm's foundation, rooted in connection contribution evaluation and brain-inspired pruning, is inherently adaptable to a wide range of neural network architectures and application domains.

5. CONCLUSION

This study introduced a dynamic pruning method inspired by synaptic pruning in the human brain to optimize deep neural network architectures. By continuously monitoring the real-time contribution of connections during training, the method preserves important neurons and gradually eliminates redundant ones. Simulation results demonstrated improved or preserved accuracy, significant network compression, and faster training times. Additionally, the method showed robustness against noisy data, highlighting its practical applicability. A key advantage of our method is its ability to enhance generalization by reducing network complexity, thereby mitigating overfitting. The minimal gap between training and testing performance across different datasets confirms this capability. Furthermore, comparative analysis indicated that our approach outperforms both conventional pruning techniques and some recent state-of-the-art methods, in terms of accuracy and compression rates. Overall, the findings demonstrate that the proposed pruning strategy is efficient for optimizing neural networks. Future work will explore its extension to more complex architectures and broader application domains, along with further validation on additional benchmark datasets.

6. REFERENCES

[1] Z. Allen-Zhu, Y. Li, and Y. Liang. (2019, Dec.). Learning and Generalization in Overparameterized Neural Networks, Going Beyond Two Layers. *Advances in Neural Information Processing Systems*. [Online]. *32*, pp. 1–13.

[2] C. Cortes, X. Gonzalvo, V. Kuznetsov, M. Mohri, and S. Yang. (2017, Jul.). Adanet: Adaptive Structural Learning of Artificial Neural Networks. Presented at International conference on machine learning (ICML). [Online].

[3] Y. Chauvin. (1990, Feb.). Generalization Performance of Overtrained Back-Propagation Networks. Presented at European Association for Signal Processing Workshop. [Online]. pp. 45-55. Available: https://doi.org/10.1007/3-540-52255-7_26

[4] G. G. Towell, M. W. Craven, and J. W. Shavlik. (1991, Jun.). Constructive Induction in Knowledge-Based Neural Networks. Presented at Proceedings of the Eighth International Conference (ML91), pp. 213-217. [Online]. Available: https://doi.org/10.1016/B978-1-55860-200-7.50046-5

[5] Y. Huang, Y. Cheng, A. Bapna, O. Firat, D. Chen, M. Chen, H. Lee, J. Ngiam, Q. V. Le, and Y. Wu. (2019, Dec.). GPipe: Efficient Training of Giant Neural Networks Using Pipeline Parallelism. *Advances in Neural Information Processing Systems*. [Online]. *32*, pp. 1–13.

[6] V. Sze, Y. H. Chen, T. J. Yang, and J. S. Emer. (2017, Dec.). Efficient Processing of Deep Neural Networks: A Tutorial and Survey. *Proceedings of the IEEE*. [Online]. *105(12)*, pp. 2295–2329. Available: https://doi.org/10.1109/JPROC.2017.2761740

[7] S. A. Janowsky. (1989, Jun.). Pruning Versus Clipping in Neural Networks. *Physical Review A*. [Online]. *39(12)*, p. 6600.

[8] G. Chechik, I. Meilijson, and E. Ruppin. (1999, Jun.). Neuronal Regulation: A Biologically Plausible Mechanism for Efficient Synaptic Pruning in Development. *Neurocomputing*. [Online]. *26–27*, pp. 633–639. Available: https://doi.org/10.1016/S0925-2312(98)00161-1

[9] A. Pascual-Leone, A. Amedi, F. Fregni, and L. B. Merabet. (2005, Jul.). The Plastic Human Brain Cortex. *Annual Review of Neuroscience*. [Online]. *28(1)*, pp. 377–401. Available: https://doi.org/10.1146/annurev.neuro.27.070203.144216

[10] C. A. Mangina and E. N. Sokolov. (2006, Apr.). Neuronal Plasticity in Memory and Learning Abilities: Theoretical Position and Selective Review. *Psychophysiology*. [Online]. *60(3)*, pp. 203–214. Available: https://doi.org/10.1016/j.ijpsycho.2005.11.004

[11] M. V. Johnston, A. Ishida, W. N. Ishida, H. B. Matsushita, A. Nishimura, and M. Tsuji. (2009, Jan.). Plasticity and Injury in the Developing Brain. *Brain and Development*. [Online]. *31(1)*, pp. 1–10. Available: https://doi.org/10.1016/j.braindev.2008.03.014

[12] G. Hinton, O. Vinyals, and J. Dean. (2015, Mar.). Distilling the Knowledge in a Neural Network. *arXiv preprint*. [Online]. Available: https://doi.org/10.48550/arXiv.1503.02531

[13] G. Chechik, I. Meilijson, and E. Ruppin. (1998, Oct.). Synaptic Pruning in Development: A Computational Account. *Neural Computation*. [Online]. *10(7)*, pp. 1759–1777. Available: https://doi.org/10.1162/089976698300017124

[14] S. Han, J. Pool, J. Tran, and W. Dally. (2015, Dec.). Learning Both Weights and Connections for Efficient Neural Network. *Advances in Neural Information Processing Systems*. [Online]. *28*, pp. 1–13.

[15] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz. (2019, Jun.). Importance Estimation for Neural Network Pruning. *Presented at Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11264–11272. [Online].

[16] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz. (2016, Nov.). Pruning Convolutional Neural Networks for Resource Efficient Inference. *arXiv preprint*. [Online]. Available: https://doi.org/10.48550/arXiv.1611.06440

[17] Y. LeCun, J. Denker, and S. Solla. (1989, Dec.). Optimal Brain Damage. *Advances in Neural Information Processing Systems*. [Online]. *2*, pp. 598–605.

[18] B. Hassibi and D. Stork. (1992, Dec.). Second Order Derivatives for Network Pruning: Optimal Brain Surgeon. *Advances in Neural Information Processing Systems*. [Online]. 5, pp. 164–171.

[19] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang. (2019, Jun.). Filter Pruning via Geometric Median for Deep Convolutional Neural Network Acceleration. Presented at Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp. 4340–4349. [Online].

[20] R. Yu, A. Li, C. F. Chen, J. H. Lai, V. I. Morariu, X. Han, M. Gao, C. Y. Lin, and L. S. Davis. (2018, Jun.). NISP: Pruning Networks Using Neuron Importance Score Propagation. Presented at Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp. 9194–9203. [Online].

[21] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf. (2016, Aug.). Pruning Filters for Efficient ConvNets. *arXiv preprint*. [Online]. Available: https://doi.org/10.48550/arXiv.1608.08710

[22] Y. He, G. Kang, X. Dong, Y. Fu, and Y. Yang. (2018, Aug.). Soft Filter Pruning for Accelerating Deep Convolutional Neural Networks. *arXiv preprint*. [Online]. Available: https://doi.org/10.48550/arXiv.1808.06866

[23] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. (2014, Jun.). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*. [Online]. *15*, pp. 1929–1958.

[24] L. Wan, M. Zeiler, S. Zhang, Y. LeCun, and R. Fergus. (2013, Jun.). Regularization of Neural Networks Using DropConnect. Presented at International Conference on Machine Learning (ICML), pp. 1058–1066. [Online].

[25] X. Sun, X. Ren, S. Ma, and H. Wang. (2017, Jun.). MeProp: Sparsified Back Propagation for Accelerated Deep Learning with Reduced Overfitting. *arXiv preprint*. [Online]. Available: https://doi.org/10.48550/arXiv.1706.06197

[26] P. J. Angeline, G. M. Saunders, and J. B. Pollack. (1994, Jan.). An Evolutionary Algorithm That Constructs Recurrent Neural Networks. *IEEE Transactions on Neural Networks*. [Online]. *5(1)*, pp. 54–65. https://doi.org/10.1109/72.265960

[27] X. Yao and Y. Liu. (1996, Mar.). Evolving Artificial Neural Networks Through Evolutionary Programming. Presented at Proceedings of the 5th Annual Conference on Evolutionary Programming, pp. 257–266. [Online].

[28] J. C. Park and S. T. Abusalah. (1997). Maximum Entropy: A Special Case of Minimum Cross-Entropy Applied to Nonlinear Estimation by an Artificial Neural Network. *Complex Systems*. [Online]. *11*, pp. 289–307.

[29] E. Vonk, L. C. Jain, and R. Johnson. (1995, Dec.). Using Genetic Algorithms with Grammar Encoding to Generate Neural Networks. *Presented at Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1928–1931. [Online]. Available: https://doi.org/10.1109/ICNN.1995.488965

[30] I. Ioan, C. Rotar, and A. Incze. (2004, Sep.). The Optimization of Feed-Forward Neural Network Structure Using Genetic Algorithms. Presented at Proceedings of the International Conference on Theory and Applications of Mathematics and Informatics (ICTAMI), Thessaloniki, pp. 223–234. [Online].

[31] F. Zhao, T. Zhang, Y. Zeng, and B. Xu. (2017, Oct.). Towards a Brain-Inspired Developmental Neural Network by Adaptive Synaptic Pruning. Presented at Proceedings of the International Conference on Neural Information Processing, pp. 182–191. [Online]. Available: https://doi.org/10.1007/978-3-319-70093-9_19

[32] A. Ahmadi and B. Mashoufi. (2012, May). New Optimized Approach for Artificial Neural Networks Training Using Genetic Algorithms and Parallel Processing. *International Review on Computers and Software*. [Online]. *7(5)*, pp. 2232–2238.

[33] J. Frankle and M. Carbin. (2019, Mar.). The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. *arXiv preprint*. [Online]. Available: https://doi.org/10.48550/arXiv.1803.03635

[34] N. Lee, T. Ajanthan, and P. H. Torr. (2019, Feb.). SNIP: Single-Shot Network Pruning Based on Connection Sensitivity. *arXiv preprint*. [Online]. Available: https://doi.org/10.48550/arXiv.1810.02340

[35] C. Wang, G. Zhang, and R. Grosse. (2020, Apr.). Picking Winning Tickets Before Training by Preserving Gradient Flow. *arXiv preprint*. [Online]. Available: https://doi.org/10.48550/arXiv.2002.07376

[36] V. Sanh, T. Wolf, and S. Ruder. (2020, May). Movement Pruning: Adaptive Sparsity by Fine-Tuning. *arXiv preprint*. [Online]. Available: https://doi.org/10.48550/arXiv.2005.07683

[37] S. Han, J. Pool, J. Tran, and W. J. Dally. (2015, Dec.). Learning Both Weights and Connections for Efficient Neural Networks. *Advances in Neural Information Processing Systems*. [Online]. *28*, pp. 1135–1143. Available: https://doi.org/10.48550/arXiv.1506.02626

[38] R. Morini, M. Bizzotto, F. Perrucci, F. Filipello, and M. Matteoli. (2021, Feb.). Strategies and Tools for Studying Microglial-Mediated Synapse Elimination and Refinement. *Frontiers in Immunology*. [Online]. *12*, no. 640937. Available: https://doi.org/10.3389/fimmu.2021.640937

[39] R. P. Guidorzi, M. P. Losito, and T. Muratori. (1982, Oct.). The Range Error Test in the Structural Identification of Linear Multivariable Systems. *IEEE Transactions on Automatic Control*. [Online]. *27(5)*, pp. 1044–1054. Available: https://doi.org/10.1109/TAC.1982.1103068

[40] E. Abdul Jaleel and K. Aparna. (2015). Identification of Ethane-Ethylene Distillation Column Using Neural Network and ANFIS. Presented at Proceedings of the 5th International Conference on Advances in Computing and Communications (ICACC). [Online].

Journal of Computer and Knowledge Engineering, Vol. 8, No. 2, 2025. (55-74)

Ferdowsi University of Mashhad

Journal of Computer and Knowledge Engineering

https://cke.um.ac.ir

Information and Communication Technology Association of Iran

Hybrid Filter-Wrapper Feature Selection using Modified Flower Pollination Algorithm*

Research Article

Mohammad Ansari Shiri[1], Najme Mansouri[2]

10.22067/cke.2025.89191.1124

Abstract *A major challenge in machine learning and data science is feature selection. Feature selection involves selecting the optimal (or suboptimal) subset of features to derive useful conclusions from a dataset based on the relevant information contained in those features. The Flower Pollination Algorithm (FPA) is a metaheuristic algorithm developed recently based on flower pollination. In this paper, we propose a new type of binary FPA, called the Filter-Wrapper Modified Binary FPA (FWMBFPA), which aims to improve convergence rate and solution quality by combining filter and wrapper advantages. Using FWMBFPA, the exploration process is directed toward specific search areas by extracting the features of existing solutions. 18 UCI datasets are used to evaluate the performance of the method. FWMBFPA generally performs better than the other algorithms in terms of average classification accuracy. FWMBFPA achieves the highest classification accuracy with the smallest number of selected features when compared to other algorithms when dealing with datasets with a large number of features.*

Key Words Feature selection, Flower Pollination Algorithm, Filter, Wrapper

1. INTRODUCTION

A broad range of fields can now access large amounts of data thanks to the advanced tools for collecting data. Data mining and machine learning tasks are greatly affected by data dimensionality [1]. Features increase in number as data dimensions increase. Thus, feature selection is used for selecting the optimal feature subset [2]. Data is reduced in dimensionality through a preprocessing step called Feature Selection (FS), which decreases learning times and eliminates irrelevant or redundant data points. The performance of supervised and unsupervised FS is degraded by redundant and irrelevant features, both of which add complexity [3], [4], [5]. A feature selection process can generally be divided into four steps: generation, evaluation, stopping conditions, and verification. It is crucial to evaluate the subset of features effectively [6]. The first step involves generating a subset of candidate features. In the second step, an evaluation indicator is used to assess the quality of the feature subset. When the process stops at step three, a feature subset meeting the stop criteria is output. FS does not directly involve the last step, but checks that the final feature subset is valid [7].

Researchers have been studying FS methods for decades. The methods are broadly categorized into four types: filter models, wrapper models, embedded models, and hybrid models [8], [9]. It does not require a learning algorithm to evaluate the filter model because it is based on the features' properties. The process is quicker and more efficient. However, since the filter model doesn't take into account the learning algorithm, some irrelevant features could be deleted, while some redundant ones would be retained. Filter approaches generally don't provide as high a classification accuracy as wrappers, so their feature subsets are generally less accurate. The wrapper model uses the classification algorithm for evaluating the results, which increases accuracy. Classification algorithms will need to be learned and verified. A large amount of data has a limited amount of running time, so the algorithm cannot evaluate each combination of features exhaustively. For this reason, heuristic optimization algorithms must be applied to help select feature subsets [10]. By using the original data directly for training, the embedded model constructs a classifier using only the optimal subset of features. These methods, however, take time and require knowledge of background parameters. To balance the algorithm's performance and time, the hybrid model combines the advantages of filter and wrapper models. It

* Manuscript received 2024 August 2, Revise 2025 April 16, Accepted 2025 June 28.

[1] M.Sc. Student Department of Computer Science, Shahid Bahonar University of Kerman, Kerman, Iran.
Email: mansari@math.uk.ac.ir
[2] Corresponding Author. Ph.D. Instructor Department of Computer Science, Shahid Bahonar University of Kerman, Kerman, Iran,
Email: n.mansouri@uk.ac.ir

makes sense to develop a hybrid-based FS algorithm [11].

It is challenging to find a subset that is (nearly) optimal from the original set. In the past two decades, metaheuristics have proved to be highly reliable solutions to a wide range of optimization problems, including engineering design, machine learning, data mining, scheduling, and production problems [12]. Researchers have investigated metaheuristics in the field of feature selection [13]. It is NP-hard to solve FS with N features since there are 2N solutions to consider. When it comes to FS methods, there are three main search strategies:

- A complete (brute-force) search that generates all possible solutions before selecting the best
- Choosing subsets randomly and hoping to find the best subset
- Random search methods guided by heuristic information.

It is impractical to use complete and random methods with FS when dealing with medium- and large-scale datasets, and a random search becomes more complete when dealing with such datasets. By combining local and global search methods, heuristic search methods produce good (not always best) solutions within a reasonable timeframe [14]. It has become more common to use metaheuristic algorithms to mimic the evolution of living creatures. It is possible to find optimal global solutions using metaheuristic methods. They are more efficient than classical algorithms at solving complex, nonlinear, and indeterminate problems. When new data enters or the environment changes, metaheuristic algorithms don't have to restart.

Simple, independent of the problem, flexible, and gradient-free characteristics are some of the advantages of meta-heuristics. Physical phenomena, animal behavior, and evolutionary concepts are common inspirations for meta-heuristics. Additionally, meta-heuristics are independent of the problem's nature, since they use a stochastic approach, which means they don't require derivative information. Unlike mathematical programming, which requires detailed knowledge about the mathematical problem, this program requires no prior knowledge. Because of their independence from the nature of the problem, they are a suitable tool for solving optimization problems without being concerned about the nonlinearity of the search space. Additionally, the algorithms' flexibility allows them to solve virtually any optimization problem without changing their structure significantly. This feature allows them to be a potential candidate for a user-friendly optimizer since they approach the problem as a black box with input and output states. Furthermore, they are mainly based on stochastic operators, unlike mathematical methods, which are deterministic. Therefore, conventional deterministic methods are less likely to lead to local optima. Their independence from the initial guess also enables them to be more flexible. Selecting features based on evolutionary algorithms can reduce the amount of time consumed and make classifications more accurate. Any problem that can be formulated can be solved using them when they are integrated with other optimization techniques. While these algorithms use mathematical formulas to solve problems, they are very fast and accurate [5], [15], [16].

Metaheuristic algorithms have two core concepts: exploration and exploitation. In exploration, the problem space is searched without concern for the results, while in exploitation, the focus is on the results. These capabilities need to be balanced to perform optimally in problem-solving [17], [18]. The exploration phase generally benefits from population-based algorithms. In addition, local search algorithms are typically used during the exploitation phase, since they can condense and find the most suitable solutions close to the original ones [19]. The purpose of this study is to combine the benefits of both a filter method and a wrapper method by using a modified flower pollination optimizer.

This study proposes a hybrid approach to address the crucial challenge of feature selection. This paper makes the following contributions:

- Proposing a novel FWMBFPA that effectively combines the advantages of filter and wrapper methods to improve feature selection performance.
- Developing a modified binary version of the flower pollination algorithm specifically tailored for feature selection problems, enhancing its search capabilities and convergence properties.
- Integrating a two-phase filtering mechanism based on Spearman correlation and relevance to efficiently reduce the dimensionality of the dataset and eliminate redundant and irrelevant features before the wrapper phase.
- Conducting extensive experiments on 18 diverse UCI datasets to demonstrate the superior performance of the proposed FWMBFPA in terms of average classification accuracy and achieving a significantly smaller number of selected features compared to several state-of-the-art metaheuristic algorithms.

The remainder of the article is organized as follows. An overview of flower pollination algorithms is provided in Section 2. Related works are highlighted in Section 3. Section 4 presents the proposed method. A comparison and evaluation of the proposed method is presented in Section 5, and the conclusion and future work are provided in Section 6.

2. PRELIMINARIES

In this section, the concepts used are explained.

2.1. Flower Pollination in Optimization Context: Nature's Inspiration

The majority of plants around the world are flowering plants, where pollination is their primary means of reproduction. During pollination, pollen is transferred from one flower to another by wind, insects, butterflies, bees, birds, and bats. Several evolutionary processes have evolved to ensure pollination by producing nectar to attract pollinators. Additionally, some pollinators and plant species, such as hummingbirds and ornithophilous flowers, contribute to flower constancy in co-evolution.

There are two basic types of pollination: biotic and abiotic [20], [21].

Biotic pollination: Pollination is primarily accomplished by biotic pollinators, such as insects, birds, and others. Pollination of flowering plants by this method is used by almost 90% of them. Pollen can travel a long distance as pollinators move at different paces and speeds. It is also possible to consider pollination with such properties to be global pollination. This action can be equated to a global search if pollen is encoded as a solution vector.

Abiotic pollination: In addition to pollination by pollinators, abiotic pollination is also called self-pollination. This form of pollination is estimated to be used by about 10% of floral plants. In local and self-pollinated plants, pollination is usually achieved by wind and diffusion. This type of motion is typically short in distance, making it suitable for use in local searches.

Flower constancy: A partnership between plants and pollinators, such as hummingbirds, can be beneficial for both parties to save energy and achieve success. The result is flower constancy. A flower plant evolves so that pollinators are rewarded with nectar from a fixed set of flower types, while pollinators spend no energy exploring new flower types. To maximize pollinator reproduction by encouraging frequent visits by them [20], [21]. The flower pollination algorithm was developed using the main characteristics of pollination.

2.2. Flower Pollination Algorithm

Based on mimicking flower pollination, Yang [22] proposed the flower pollination algorithm as a metaheuristic optimization algorithm.

To ensure the quality of the search, FPA mixes exploitation and exploration randomly. As a result of FPA, the following idealized principles are followed [22]:

- Rule 1: Through Lévy flight, biotic cross-pollination acts as a global search.
- Rule 2: Local searches are abiotic and self-pollinated.
- Rule 3: The similarity between two flowers can lead to flower constancy.
- Rule 4: Local and global searches are switched randomly ∈ [0, 1].

Initially, a random population is generated, and then the optimal solution is determined by evaluating the data. A new solution can be calculated by determining the pollination type according to a predetermined probability p (Rule 4). Assuming r is between 0 and 1, global pollination (Rule 1) and flower constancy (Rule 3) can occur as follows if r is less than p [22], [23], [24]:

$$x_i^{t+1} = x_i^t + \gamma L(x_i^t - g_{best}) \quad (1)$$

Eq. (1) involves x_i^t as a solution, i at time t, g_{best} as the current best solution, γ as a scaling factor, and L as a step size [22], [23], [24]:

$$L(s,c) \sim \frac{\lambda \Gamma(\lambda) \sin(\pi \frac{\lambda}{2})}{\pi} \cdot \frac{c}{s^{1+\lambda}}, (s \gg s_0 > 0) \quad (2)$$

For large steps s > 0, the gamma function $\Gamma(\lambda)$ is valid. The tail amplitude of the distribution is controlled by c, which is 1 in the proposed FPA. According to Rule 2, local pollination (Rule 2) and flower constancy can be expressed as follows [22], [23], [24]:

$$x_i^{t+1} = x_i^t + \varepsilon(x_j^t - x_k^t) \quad (3)$$

The pollens x_j^t and x_k^t come from flowers of the same plant species. As a result, flowers remain constant in a limited area. ε comes from a uniform distribution in [0,1], x_j^t and x_k^t give a local random walk if they are from the same species. The pollination of flowers can take place locally as well as globally. A nearby flower patch or flowers in a neighboring neighborhood are more likely to be pollinated by local flower pollen than those far away. The switch probability (Rule 4) or proximity probability p is used to switch between intensive local pollination and common global pollination. To determine the most appropriate parameter range, it is possible to use p = 0.5 as an initial value and then do a parametric study to determine the most appropriate parameter range. Algorithm 1 shows the pseudo-code of the FPA [22].

3. RELATED WORKS

In classification, feature selection is crucial. Recently, several algorithms have been developed for solving feature selection problems. General optimization problems benefit from swarm algorithms in terms of exploitation and exploration. It is still necessary to improve the accuracy of solution selection, the speed of time consumption, and the finding of global optimums in feature selection problems. To solve these drawbacks, there are many attempts in this direction.

The Salp Swarm Algorithm (SSA) is a bio-inspired algorithm designed to optimize a system using the swarming mechanisms of Salps [25]. Using Salp's swarm algorithm, Hegazy et al. [26] overcame the low convergence rate and avoided getting stuck in a local optimum. Twenty-seven datasets are used to evaluate the performance of CSSA when it is combined with the K-nearest neighbor classifier to solve the feature selection problem.

Using a wrapper approach, Naik et al. [27] identified the relevant subset of features for machine learning tasks. The Binary Bat algorithm is used to select a set of features, and a novel fitness function is implemented using One-pass Generalized Classifier Neural Networks (OGCNN). This fitness function takes into account the entropy of sensitivity, specificity, classifier accuracy, and fraction of selected features. Using four classifiers (Radial Basis Function Neural Networks, Probabilistic Neural Networks, Extreme Learning Machines, and OGCNNs), fitness functions are also compared on six publicly available datasets. Using one-pass classifiers is more efficient from a computational standpoint. Results indicated that OGCNN performed well in most cases when combined with the novel fitness function.

Algorithm 1. Flower Pollination Algorithm Pseudo-Code

1: Objective min or max **f(x)**, x = (x_1, …, x_d)
2: Initialize a population of n flowers/pollen with random solutions
3: Find the best solution g_{best} in the initial population
4: Define a switch probability p ∈ [0, 1]
5: **While** (stopping criterion not satisfied) **do**
6: **For** i = 1: n (all *n* flowers in the population)
7: **If** rand () < p
8: Draw a step vector *L* that obeys a Levy distribution
9: Global pollination: $x_i^{t+1} = x_i^t + \gamma L(x_i^t - g_{best})$
10: **else**
11: Select two random solutions x_j^t and x_k^t
12: Local pollination: $x_i^{t+1} = x_i^t + \varepsilon(x_j^t - x_k^t)$
13: **end if**
14: Evaluate new solutions
15: If new solutions are better, update them in the population
16: **end for**
17: Keep the current best solution
18: **end while**

Gao et al. [28] presented two algorithms for optimizing binary balance and selecting the best feature subset for classification problems. Equilibrium optimizers (EOs) are optimization algorithms based on physics [15]. In order to estimate dynamic and equilibrium states, it is based on models of controlled volumetric mass balance. First, the BEO-S and BEO-V algorithms map continuous EOs to discrete types. To determine the position of the optimal solution, the position vector (BEO-T) is used. A comparison of the proposed algorithm with other advanced FS algorithms is conducted on 19 well-known UCI datasets. Experimental results proved that BEO-V2 outperforms other state-of-the-art metaheuristic algorithms in terms of performance measures among the proposed binary EO algorithms.

The Grasshopper Optimization Algorithm (GOA) is an algorithm that mimics grasshopper migration and hunting in nature [29]. As a result of the low diversity of agents, this method tends to stagnate or become immature. Using SCGOA, Zhao et al. [30] proposed a new GOA with exploration and exploitation features to improve GOA's ability to handle a wide variety of situations. As a first step, trigonometric substitution is used to disturb people's position vector updates (evolution) to balance the exploration and exploitation stages in the proposed SCGOA. A Cauchy mutation-based strategy increases the diversity of the locust population and prevents stagnation. The Cauchy mutation ensures the diversity of locust populations. A comparison of SCGOA with several well-known meta-heuristic algorithms was conducted using the latest IEEE CEC2017 benchmark functions. The proposed SCGOA is superior to its rivals based on some extensive analysis results. The results of the study demonstrated that SCGOA was superior to some existing algorithms when applied to four engineering design problems based on Cauchy mutations. Several feature selection datasets were also handled using the binary version of Cauchy mutation-based SCGOA. Binary version of GOA outperforms original GOA and other optimization algorithms when it comes to classifying, having fewer errors, and fewer features.

In 2015, Duggan and Olmes [31] developed the Vortex Search Algorithm (VSA), a meta-heuristic algorithm based on the vortex phenomenon. Using chaos theory, Gharehchopogh et al. [32] overcome the entrapment of local optima, obtain the optimal feature set with maximum accuracy and minimum number of features. The proposed method considers various chaotic maps to improve the VSA operators and control both exploration and exploitation. Datasets from 24 UCI standards were used to evaluate this method's performance. This method was also evaluated as a Feature Selection (FS) approach. Based on simulation results, chaotic maps (especially the Tent map) can improve the performance of the VSA. In addition, it was demonstrated that the proposed method provided the best accuracy and the smallest number of features for determining the optimal feature subset. As compared to other algorithms, the proposed method performed better in the real application.

Using the firefly algorithm (FA) previously developed by Bacanin et al. [33], a new feature selection problem was addressed. Compared with the original FA, the proposed method performs much better in limited and practical terms. After validating the method on unconstrained benchmarks, 21 standard datasets from the University of California, Irvine (UCI) were used for feature selection. Furthermore, a new COVID-19 dataset was used in the present study to predict the health of patients, as well as a microcontroller microarray dataset. Based on the results of all practical simulations, we can certify the robustness and efficiency of the proposed algorithm when it comes to convergence, quality of solutions, and classification accuracy. In more detail, the proposed approach outperformed other competitor methods on 13 out of 21 datasets.

TABLE1
Related works on feature selection

Ref.	Year	FS method	Advantages	Dataset used	Filter/Wrapper
Hegazy et al. [26]	2019	CSSA	Fewer parameters, simpler to implement	27	Wrapper
Naik et al. [27]	2020	OGCNN	Implementation using four classifiers, high accuracy	6	Wrapper
Gao et al. [28]	2020	BEO-S BEO-V	Extensive exploration and exploitation ability to change the solution at random	19	Wrapper
Zhao et al. [30]	2022	SCGOA	Enhance GOA's capability to handle diverse situations, avoid stagnation and laziness	-	Wrapper
Gharehchopogh et al. [32]	2022	VSA	Involving chaotic maps in VSA prevents local optima	24	Wrapper
Bacanin et al. [33]	2023	FA	High convergence speed	22	Wrapper

Although numerous metaheuristic algorithms have been used to select features, a critical review of existing literature reveals limitations and a research gap that this study aims to fill. The majority of previous studies used wrapper-based approaches (as summarized in Table 1), which, though often providing high accuracy, can be computationally expensive and suffer from scalability issues when dealing with high-dimensional datasets due to the lack of a feature reduction step at the outset. In these methods, feature subsets are evaluated using a learning algorithm, which becomes time-consuming as the number of features increases.

Additionally, purely filter-based methods are computationally efficient, but evaluate features independently or based on intrinsic properties, potentially overlooking feature interactions and their influence on specific learning algorithms. Despite the fact that hybrid methods combine filter and wrapper approaches, there is still a need for more efficient and robust hybrid algorithms that can effectively balance the computational speed of filters with the accuracy of wrappers, especially in complex, high-dimensional datasets.

It is therefore necessary to develop a hybrid filter-wrapper feature selection algorithm that not only integrates the strengths of both paradigms but also enhances the optimization engine to ensure efficient exploration and exploitation of the search space. By introducing an initial filtering phase that handles high dimensionality and by integrating modifications to the Flower Pollination Algorithm's search mechanism within the wrapper phase, the FWMBFPA is proposed as a means of bridging this gap.

4. PROPOSED METHOD

The Flower Pollination Algorithm (FPA), introduced by Yang [22], is a nature-inspired metaheuristic algorithm that has demonstrated promising performance in solving various optimization problems. FPA is based on the fascinating process of flower pollination, incorporating both global pollination (biotic and cross-pollination via Lévy flight) and local pollination (abiotic and self-pollination). As a result of this inherent duality, FPA is able to balance exploration (searching diverse areas of the search space) and exploitation (refining potential solutions).

In feature selection, the goal is to find a subset of features that maximizes classification accuracy while minimizing the number of selected features. This NP-hard problem is well suited to metaheuristic algorithms. The ability of FPA to balance global and local search makes it an ideal candidate for navigating the complex and high-dimensional binary search space of feature selection. With its structure of updating based on the best solution and random interactions on the local level, it provides a solid foundation for adapting and modifying. Due to FPA's demonstrated effectiveness in optimization and its intrinsic mechanisms for exploration and exploitation, which are crucial for effectively searching the feature subset space, FPA was chosen for the wrapper phase. As a result, its structure allows for targeted modifications, such as the Modified FPA (MFPA) within our proposed FWMBFPA, that further enhance its performance. The framework of the proposed method is presented in this section, which is divided into two phases.

4.1. Filter phase

In order to handle high-dimensional data efficiently, the initial feature set needs to be reduced during this filter phase. This process begins with identifying and handling redundant features.

4.1.1. Redundancy computation

In the initial step of the filter phase, we address feature redundancy in order to reduce dimensionality. In order to accomplish this, the correlation between features is computed and analyzed.

In redundancy, two or more attributes are dependent on each other. MI measures how much a feature depends on a Subset (S) of features. Features that set symmetry, non-linearity, non-negativeness, and non-decreasing properties are observable as features are added. This measure does not indicate which features of S are redundant. A Markov blanket and total correlation are both useful over time measures that reduce redundancy. To assess numerical characteristics and subject matter knowledge, data-driven correlation analysis is useful. Data-driven methodologies

can be used to calculate correlation coefficients between two features quickly. A highly correlated trait must have a correlation coefficient that exceeds a certain threshold to be eligible to calculate the Spearman correlation coefficient. Spearman correlation coefficients were used to estimate the correlation between two features. Correlation analysis uses Eq. (10) as an alternative stimulus condition. In addition to linear correlations, SCC can also measure nonlinear correlations. The SCC measures the degree to which two features are closely related. SCC values increase with stronger correlations. Feature f_i and feature f_j should not have an SCC greater than k_1 when there is a high correlation between them [34], [35], [36].

$$corr(f_i, f_j) = |SCC(f_i, f_j)|, n \geq k_1 \quad (4)$$

TABLE 2
Levels of relevance for feature fi

Relevance level	**Condition**	**Probabilistic approach**	**Mutual information approach**
Strongly relevant	$\nexists$	$p(C\|f_i, \neg f_i) \neq p(C\|\neg f_i)$	$I(f_i; C\|f_i) \succ 0$
Weakly relevant	$\exists S \subset \neg f_i$	$p(C\|f_i, \neg f_i) \neq p(C\|\neg f_i) \wedge p(C\|f_i, S) \neq p(C\|S)$	$I(f_i; C\|f_i) \succ 0 \wedge I(f_i; C\|S) \succ 0$
Irrelevant	$\exists S \subset \neg f_i$	$p(C\|f_i, S) \neq p(C\|S)$	$I(f_i; C\|S) \succ 0$

4.1.2. Relevance computation

Generally, an attribute is relevant if it provides information on a class tag attribute alone (C) or if it provides information when combined with another variable.

Weakly associated features, highly associated features, and unrelated features have been used to define associations. When a feature is strongly related to C, it cannot be replaced with another feature without removing its information. A weakly associated feature provides information about C, but can be replaced by another without losing any information. It is possible to lose information about C when you remove irrelevant features from it. In Table 2. the relevance levels of feature f_i are shown [34], [35], [36].

4.2. Wrapper phase

In the wrapper phase, the optimal feature subset is selected using a modified optimization algorithm, described in this subsection. In the wrapper phase, an evolutionary search approach is used to select the optimal subset based on the reduced and more relevant feature set obtained during the filter phase. The search is powered by a modified version of the flower pollination algorithm.

4.2.1. Modified Flower Pollination Algorithm

The selection of the optimal feature subset is based on an effective optimization algorithm, as described above in the wrapper phase description. For this essential search, the study uses an MFPA, described in this subsection.

Based on the clonal selection principle, the proposed MFPA modifies the standard FPA. According to experimental results, random walks produce faster convergent solutions than Levy flights in local pollination. Therefore, we replaced Levy flights with random walks. Using random uniform distributions in [0, 1], random walks are generated. A high-affinity solution is cloned proportional to its affinity before local pollination can be applied. To modify the local pollination, γ_2 was introduced as a step-size scaling factor. In a preliminary parametric study, it was found that γ_2= 3 is effective for all test cases. At iteration *t*, the MFPA selects the top 14 solutions from a population *Pop* and clones each solution proportionally based on its fitness. Cloned solutions are likely to be exploited, so to avoid getting stuck in local minima, the algorithm checks a value not greater than 10e^6 for 100 successive iterations. In such a case, the entire population Pop is replaced by a new randomly generated one, while keeping the best solution g_{best}; this greatly increases exploration. The pseudo-code of the modified flower pollination algorithm can be seen in Algorithm 2.

4.2.2 Fitness function

In general, FS aims to minimize feature selection while maximizing classification accuracy. There is a conflict between these two objectives. It is possible to combine these two objectives into one objective problem by utilizing Eq. (5).

$$Fitness = \omega(1 - Accuracy) + (1 - \omega) \times F \quad (5)$$

A ratio *F* is computed by dividing the number of features selected by the original dimension of the dataset. The classification error rate of the selected subset of features is *(1-Accuracy).* Weight (ω) is represented by the values 0 and 1.

4.2.3. Filter-Wrapper Modified Binary Flower Pollination Algorithm (FWMBFPA)

In Eq. (6), the binary version of the algorithm is converted using sigmoid functions. As a consequence, FS can only be solved with binary values between 0 and 1. There is a binary vector for every solution, where 1 indicates that the corresponding feature has been selected, and 0 indicates that it has not been selected.

$$T(x) = \frac{1}{1+e^{-x}} \quad (6)$$

The flowchart in Fig. 1 illustrates the proposed method's overall flow. Fig. 1 shows two phases of the proposed method: filter and wrapper. Filtering is achieved using a combination of two filter methods, so that Spearman correlation between features is calculated first, a correlation limit of 0.8 is applied, and overly correlated features are discarded. By measuring the correlation between the category feature and other features, irrelevant features that were unrelated to the category feature were eliminated. Wrappers are provided with a bunch of normal, non-redundant features after the filter phase. The wrapping phase involved selecting an optimal set of features with maximum accuracy, based on the features of the modified flower pollination algorithm, which avoids the local optimum.

5. EXPERIMENTAL RESULTS AND DISCUSSION

The purpose of this section is to present the experimental setup and datasets that were used to evaluate the proposed FWMBFPA. Several experiments were conducted to evaluate and compare the performance of the proposed FWMBFPA with existing methods. These experiments are described in more detail below.

5.1. Experiment setup

The purpose of this subsection is to provide specific details of the experimental setup used to conduct the evaluation. It includes the datasets, the data splitting strategy, as well as the configuration of the classifier. With the FS method, a subset of the entire dataset was selected to evaluate KNN classifier performance. According to [37], K=5 is the recommended value for KNN classifiers.

A training dataset contains 80% of cases, while a test dataset contains 20%. The proposed method is implemented using Python3 and Matplotlib. There was 18 original UCI datasets evaluated in Table 3. Feature counts can be seen before filtering each dataset. Filtering is part of the detection phase, which detects redundancy among features, thus excluding duplicates, and ignoring features that are unrelated to category features. Additionally, the proposed method costs less to compute and can select a more accurate subset of features than existing methods. After applying the filter phase, Table 3. shows the dataset dimensions. The effect of the filter becomes more apparent on datasets with higher dimensions and more features, as shown in this table.

5.2. Evaluation of FWMBFPA and FPA

According to Table 4. BFPA and FWMBFPA were compared for classification accuracy and number of selected features across 18 datasets. As shown in Table 4, FPA had classification accuracy greater than 95% in 10 out of 18 data sets (55.55%), whereas FWMBFPA had classification accuracy greater than 95% in 12 out of 18 data sets (66.66%). Further, FWMBFPA selects fewer features in 17 data sets than BFPA and achieves the same number of features in only one data set (Zoo). To improve the performance of BFPA, the modified method that uses both the filtering and wrapping advantages has been modified and combined with the wrapper advantages. It is shown in Fig. 2 that FWMBFPA is both more accurate and has a lower mean number of selected features than BFPA.

5.3. Comparison and discussion

In this section, the results of comparing the proposed method with BFPA and 9 Binary optimization methods including Whale Optimization Algorithm (WOA), Time-Varying Salp Swarm Algorithm (TVSSA), Two-phase Mutation Gray Wolf Optimizer (TMGWO), Sine Cosine Algorithm (SCA), Jaya Algorithm (JA), Differential Evolution Algorithm (DEA), Cuckoo Search Algorithm (CSA), Bat Optimization Algorithm (BAT) and Bare Bone Particle Swarm Optimization (BBPSO) are presented.

Algorithm 2. Modified Flower Pollination Algorithm Pseudo-Code

Objective function $f(x)$, where $x= (x_1,\ldots,x_D)$ is a binary vector of dimension D.
Initialize a population of n flowers/pollen with random solutions
Find the best solution g_{best} in the initial population
Define a switch probability p $\in$ [0, 1]
While (stopping criterion not satisfied) do
    If rand () < p
        For i = 1: n (all *n* flowers in the population)
        Draw a step length vector *L* from the Lévy distribution.
            Global $x_i^{t+1} = x_i^t + \gamma L(x_i^t - g_{best})$ pollination:
        End for
    Else
        Identify the best m solutions from the current population P
        Solutions are cloned proportional to their fitness
        For each solution in clone population
            Obtain a random value $\boldsymbol{\varepsilon}$ uniformly distributed between 0 and 1
            Randomly select two distinct solutions j and k from the population P
                Local $x_i^{t+1} = x_i^t + \varepsilon(x_j^t - x_k^t)$ pollination:
        End for
    End if
    Select the best n solutions from the combined pool to form the new population
    Update the current population P with the new population P_{new}
    Identify the best solution g_{best} in the current population P
    If g_{best} doesn't improve in 100 iterations by more than 10^-6, keep g_{best} and Replace the population with a new, randomly generated binary solutions
End while
Print $\mathbf{g_{best}}$

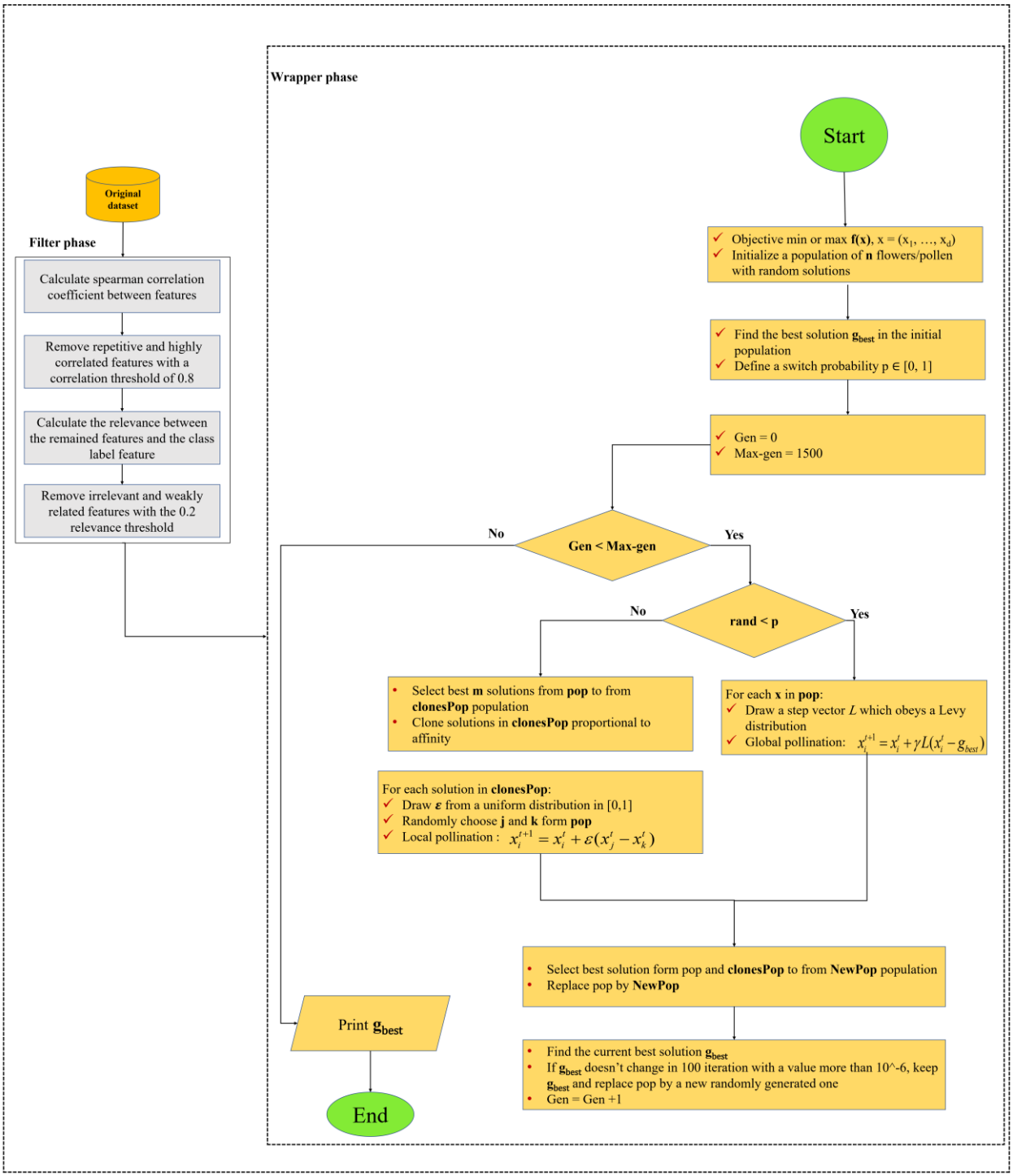


Fig. 1. Flowchart of proposed FWMBFPA

TABLE 3
Description of datasets before and after the filter phase

Dataset	**No. of sample**	**No. of features before filter**	**No. of features after filter**	**No. of class**	**Domain**
Algerian_forest_fires	244	14	8	2	Life
BreastCancer	698	11	9	2	Life
BreastEW	568	31	15	2	Life
CongressEW	434	17	15	2	Social
HeartEW	270	14	10	2	Life
Ionosphere	351	35	13	2	Physical
lung-cancer	32	57	28	2	Life
Lymphography	148	19	12	4	Life
M-of-n	1000	14	7	2	Life
Pd-speach	756	755	82	2	Life
penglung	73	326	148	7	Life
sobar-72	72	20	16	2	Physical
Sonar	208	61	35	2	Life
SpectEW	267	23	14	2	Physical
Vote	300	17	15	2	Social
Wholesale customers data	440	8	4	2	Business
Wine	178	14	12	3	Physical
Zoo	101	17	10	2	Life

TABLE 4
Classification accuracy of BFPA and FWMBFPA with selected features

	Dataset	**BFPA**		**FWBMFPA**	
		Accuracy	**No. of Features**	**Accuracy**	**No. of features**
1	Algerian forest fires	98.64	3	100	2
2	Breast cancer	100	3	100	2
3	BreastEW	96.49	14	97.07	2
4	CongressEW	96.94	7	99.23	4
5	HeartEW	83.95	5	91.35	3
6	Ionosphere	91.50	15	97.16	3
7	lung-cancer	100	23	100	6
8	Lymphography	91.11	9	93.33	5
9	M-of-n	93	10	100	6
10	Pd-speech-feature	77.53	353	84.14	14
11	penglung	95.45	149	100	20
12	Sobar72	100	4	100	3
13	Sonar	90.47	30	92.06	9
14	SpectEW	88.88	10	92.59	6
15	Vote	97.77	8	98.88	2
16	Wholesale customers data	94.69	4	94.69	1
17	Wine	98.14	6	100	2
18	Zoo	96.77	5	96.77	5

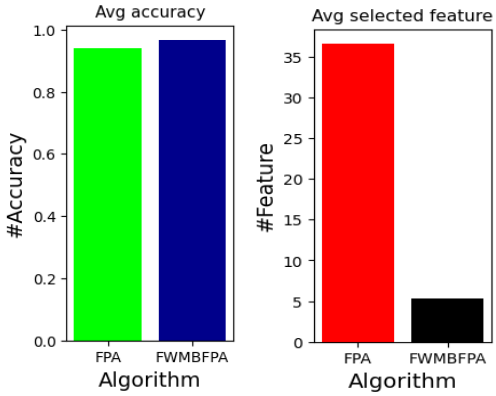


Fig. 2. Average accuracy and selected features obtained by FWMBFPA and BFPA.

TABLE 5
The worst fitness value achieved by FWMBFPA and other methods

Dataset	**FWBMFPA**	**BFPA**	**BWOA**	**TVSSA**	**TMGWO**	**BSCA**	**BJA**	**BDE**	**BCSA**	**BBAT**	**BBPSO**
Algerian forest fires	0.0158	0.0210	0.0298	0.0462	0.0439	0.0581	0.0290	0.0321	0.0030	0.0179	0.0306
Breast cancer	0.0251	0.0258	0.0380	0.0464	0.0370	0.0501	0.0332	0.0285	0.0322	0.0322	0.0144
BreastEW	0.0520	0.0686	0.0516	0.0400	0.0410	0.0339	0.0577	0.0420	0.0539	0.0751	0.0608
CongressEW	0.0420	0.0503	0.0421	0.0579	0.0396	0.0427	0.0346	0.0194	0.0346	0.0427	0.0673
HeartEW	0.1655	0.2979	0.2001	0.1635	0.2482	0.1994	0.2841	0.1757	0.2482	0.2490	0.2834
Ionosphere	0.0695	0.0884	0.1156	0.0975	0.0881	0.1071	0.1173	0.1345	0.0989	0.1270	0.1448
lung-cancer	0.1026	0.2026	0.1052	0.2026	0.2040	0.2031	0.3018	0.2028	0.3016	0.2024	0.3009
Lymphography	0.0723	0.1601	0.0907	0.2024	0.1573	0.2018	0.1364	0.1358	0.1798	0.1347	0.1810
M-of-n	0.0100	0.1414	0.1795	0.1457	0.1894	0.1851	0.1612	0.1899	0.1670	0.1792	0.0960
Pd-speech-feature	0.1701	0.2272	0.3101	0.2664	0.2886	0.2621	0.2229	0.2403	0.2321	0.2228	0.2362
Penglung	0.0948	0.0948	0.0497	0.1849	0.0944	0.0050	0.1401	0.0946	0.0497	0.0499	0.0502
Sobar72	0.0046	0.0486	0.0073	0.0513	0.0936	0.0492	0.0497	0.0497	0.0497	0.0497	0.0047
Sonar	0.1484	0.2106	0.1146	0.1305	0.1300	0.1465	0.2082	0.1935	0.1312	0.1459	0.1628
SpectEW	0.1146	0.1874	0.1512	0.1154	0.1629	0.1507	0.1666	0.1403	0.1385	0.1620	0.1629
Vote	0.0401	0.0386	0.0606	0.0276	0.0697	0.0367	0.0496	0.0593	0.0483	0.0392	0.0600
Wholesale Costomers data	0.0474	0.0867	0.0642	0.0807	0.0717	0.0703	0.0596	0.1032	0.0792	0.0910	0.0657
Wine	0.0054	0.0596	0.0412	0.0802	0.1520	0.0412	0.0596	0.0412	0.0405	0.1512	0.1489
Zoo	0.0694	0.0719	0.1327	0.0701	0.0707	0.0375	0.1008	0.1327	0.0381	0.0056	0.0350

5.3.1. Convergence rate of fitness value

Tables 5, 6, and 7 show the worst, average, and best fitness values obtained from FWMBFPA and other methods. Based on Table 5, FWMBFPA has the lowest fitness value out of 10 datasets out of 18 (55.55%), and has the worst fitness value in 8 of the 18 datasets. Bold number in all Tables shows the best performance. After the proposed method, BWOA and TVSSA have a better fitness value in the two datasets. In the entire dataset, BFPA, TMGWO, and BJA have the worst performance and worst fitness values. Table 6 shows that the proposed FWMBFPA has a significant advantage over other methods in terms of average fitness value (83.33%) in 15 datasets. Thereafter, only BCSO, BBPSO, and TMGWO had optimal fitness values in equal than one dataset, whereas BFPA did not have an optimal fitness value in any dataset. The best fitness value in 12 data sets was obtained by FWMBFPA in Table 7. (66.66%) when compared to other methods. The best fitness values in three datasets are obtained by BBAT and BBPSO, followed by TMGWO, BCSA, BDE, and BSCA, but BJA only in one dataset achieves the best value, while other methods like BFPA are not able to achieve the best value. Compared to the results derived from these three tables, FWMBFPA significantly improves BFPA's performance and is superior to other methods, and has a higher convergence rate. Fig. 3 presents the fitness values obtained using FWMBFPA and other methods for a clearer understanding of the content. To improve clarity and identification, a circular legend highlights the proposed method, shown in blue.

5.3.2. Evaluation of Classification accuracy and selected features

The accuracy of classification and the number of selected features of FWMBFPA, BFPA, and nine other methods are

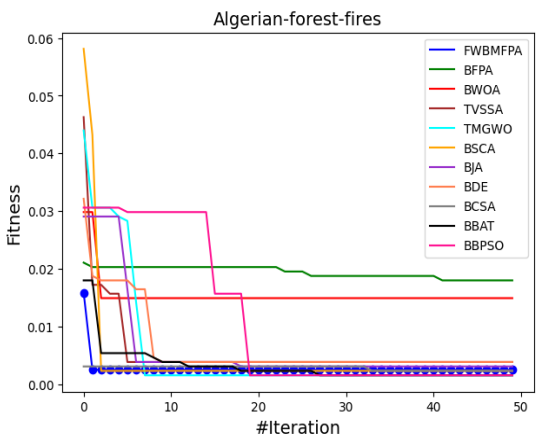
compared in this section. If the classification accuracy of a method is higher compared to other methods, it means that the method in question performed with better accuracy and had fewer errors. Among the 18 datasets analyzed in Table 8, the FWMBFPA method had the best classification accuracy in 14 datasets (77.77%) and performed better than other methods, followed by BSCA and TMGWO in 8 datasets (44.44%). Based on this Table, with better performance in two datasets, BFPA and BWOA perform the worst in terms of classification accuracy.

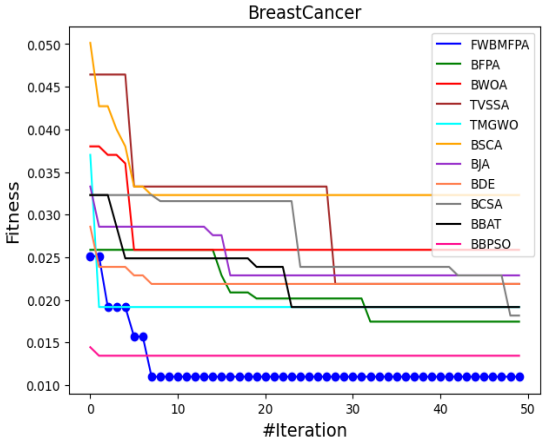
TABLE 6
The average fitness value achieved by FWMBFPA and other methods

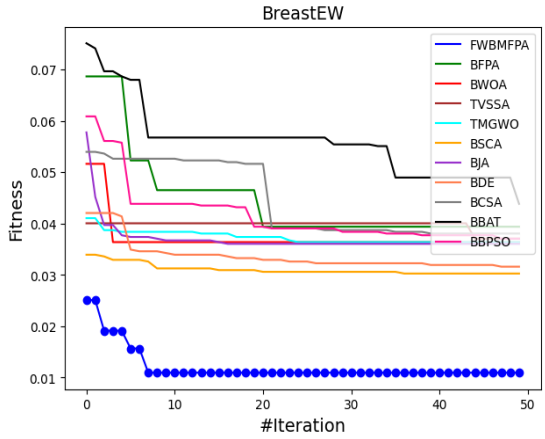
Dataset	**FWBMFPA**	**BFPA**	**BWOA**	**TVSSA**	**TMGWO**	**BSCA**	**BJA**	**BDE**	**BCSA**	**BBAT**	**BBPSO**
Algerian forest fires	0.0027	0.0193	0.0155	0.0057	0.0054	0.0042	0.0061	0.0063	0.0028	0.0031	0.0112
Breast cancer	0.0122	0.0209	0.0269	0.0295	0.0194	0.0333	0.0247	0.0221	0.0273	0.0221	0.0134
BreastEW	0.0122	0.0448	0.0373	0.0397	0.0343	0.0310	0.0370	0.0337	0.0444	0.0560	0.0417
CongressEW	0.0155	0.0375	0.0196	0.0362	0.0319	0.0265	0.0199	0.0183	0.0217	0.0229	0.0293
HeartEW	0.1009	0.2101	0.1666	0.1593	0.1483	0.1405	0.1490	0.1284	0.1567	0.1491	0.1303
Ionosphere	0.0387	0.0884	0.0822	0.0936	0.0467	0.0540	0.0658	0.0730	0.0827	0.0879	0.0849
lung-cancer	0.0229	0.1313	0.1038	0.1666	0.1091	0.1029	0.0829	0.0479	0.1130	0.0484	0.0708
Lymphography	0.0708	0.1362	0.0907	0.1679	0.1001	0.0990	0.0945	0.1124	0.1176	0.1247	0.1226
M-of-n	0.0100	0.0789	0.0782	0.1055	0.0420	0.0316	0.0146	0.0414	0.0361	0.0379	0.0102
Pd-speech-feature	0.1603	0.2271	0.2111	0.2118	0.1538	0.1817	0.2141	0.2301	0.2315	0.2267	0.2316
Penglung	0.0143	0.0750	0.0479	0.1173	0.0486	0.0014	0.0184	0.0448	0.0395	0.0493	0.0488
Sobar72	0.0022	0.0178	0.0027	0.0217	0.0240	0.0178	0.0078	0.0116	0.0297	0.0189	0.0035
Sonar	0.0935	0.1118	0.0854	0.1176	0.0793	0.0817	0.0789	0.1067	0.0706	0.1056	0.0896
SpectEW	0.0899	0.1400	0.1141	0.1098	0.0933	0.1067	0.1141	0.0931	0.1136	0.1110	0.1056
Vote	0.0202	0.0274	0.0368	0.0262	0.0333	0.0286	0.0244	0.0383	0.0323	0.0245	0.0277
Wholesale Costomers data	0.0474	0.0867	0.0499	0.0770	0.0525	0.0505	0.0527	0.0670	0.0553	0.0688	0.0555
Wine	0.0048	0.0538	0.0210	0.0633	0.0295	0.0215	0.0239	0.0228	0.0236	0.0358	0.0253
Zoo	0.0387	0.0519	0.0419	0.0405	0.0191	0.0070	0.0115	0.0330	0.0152	0.0055	0.0144

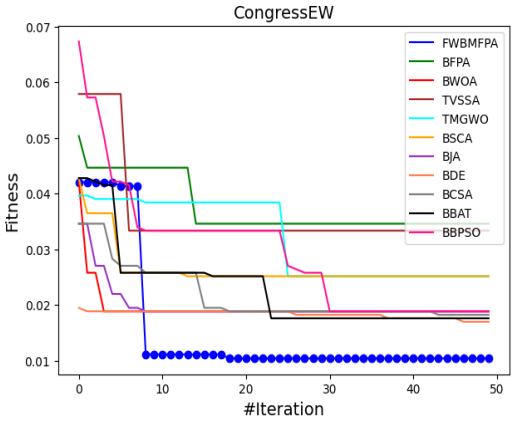
TABLE 7
The best fitness value achieved by FWMBFPA and other methods

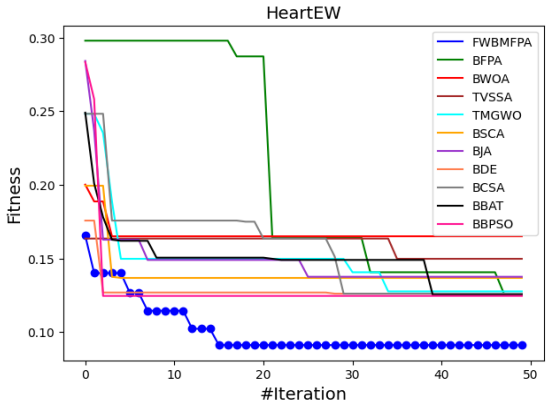
Dataset	**FWBMFPA**	**BFPA**	**BWOA**	**TVSSA**	**TMGWO**	**BSCA**	**BJA**	**BDE**	**BCSA**	**BBAT**	**BBPSO**
Algerian forest fires	0.0025	0.0179	0.0149	0.0038	0.0015	0.0023	0.0030	0.0038	0.0023	0.0015	0.0015
Breast cancer	0.0109	0.0174	0.0258	0.0218	0.0191	0.0322	0.0228	0.0218	0.0181	0.0191	0.0134
BreastEW	0.0303	0.0394	0.0364	0.0370	0.0364	0.0303	0.0360	0.0316	0.0380	0.0438	0.0370
CongressEW	0.0104	0.0346	0.0188	0.0333	0.0251	0.0251	0.0188	0.0169	0.0182	0.0176	0.0188
HeartEW	0.0911	0.1260	0.1650	0.1497	0.1276	0.1367	0.1375	0.1260	0.1260	0.1252	0.1245
Ionosphere	0.0321	0.0884	0.0764	0.0689	0.0379	0.0385	0.0604	0.0586	0.0689	0.0689	0.0578
lung-cancer	0.0026	0.1036	0.1034	0.1031	0.1002	0.0012	0.0033	0.0035	0.0035	0.0051	0.0033
Lymphography	0.0705	0.0930	0.0907	0.1144	0.0907	0.0918	0.0918	0.0913	0.0913	0.0913	0.1133
M-of-n	0.0100	0.0358	0.0325	0.0729	0.0046	0.0046	0.0046	0.0046	0.0046	0.0046	0.0046
Sobar72	0.0020	0.0057	0.0021	0.0042	0.0021	0.0036	0.0026	0.0042	0.0036	0.0047	0.0031
Sonar	0.0838	0.0967	0.0822	0.0984	0.0665	0.0519	0.0511	0.0514	0.0360	0.0989	0.0820
SpectEW	0.0779	0.1145	0.1027	0.1014	0.0891	0.1000	0.0928	0.0905	0.1023	0.1009	0.1000
Vote	0.0145	0.0270	0.0361	0.0251	0.0276	0.0238	0.0232	0.0251	0.0245	0.0141	0.0232
Wholesale Costomers data	0.0474	0.0867	0.0492	0.0746	0.0492	0.0492	0.0521	0.0642	0.0521	0.0671	0.0553
Wine	0.0036	0.0229	0.0206	0.0428	0.0046	0.0206	0.0229	0.0214	0.0214	0.0214	0.0198
Zoo	0.0374	0.0381	0.0388	0.0356	0.0037	0.0050	0.0037	0.0050	0.0031	0.0050	0.0031
Pd-speech-feature	0.1587	0.2271	0.2009	0.2099	0.1199	0.1664	0.2134	0.2182	0.2312	0.2225	0.2306
Penglung	0.0006	0.0499	0.0464	0.0488	0.0463	0.0007	0.0048	0.0038	0.0045	0.0490	0.0479

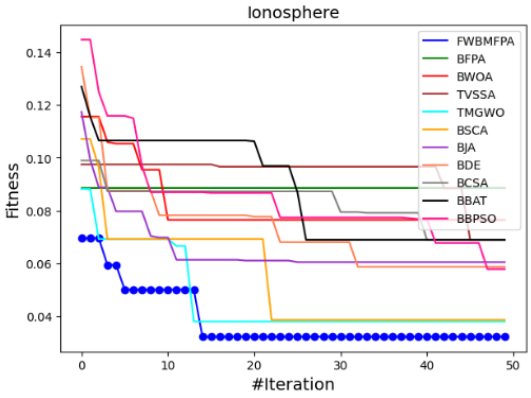

Algerian-forest-fires
Fitness
#Iteration
FWBMFPA
BFPA
BWOA
TVSSA
TMGWO
BSCA
BJA
BDE
BCSA
BBAT
BBPSO

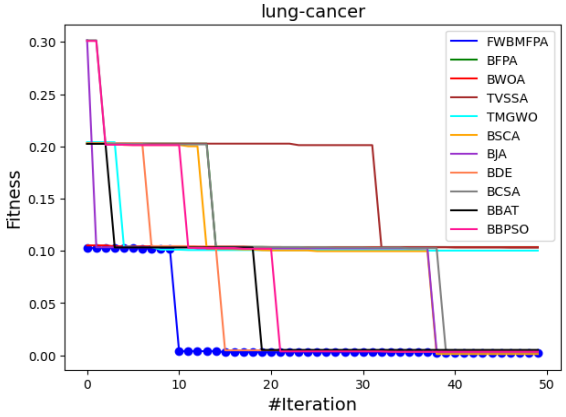

BreastCancer
Fitness
#Iteration
FWBMFPA
BFPA
BWOA
TVSSA
TMGWO
BSCA
BJA
BDE
BCSA
BBAT
BBPSO

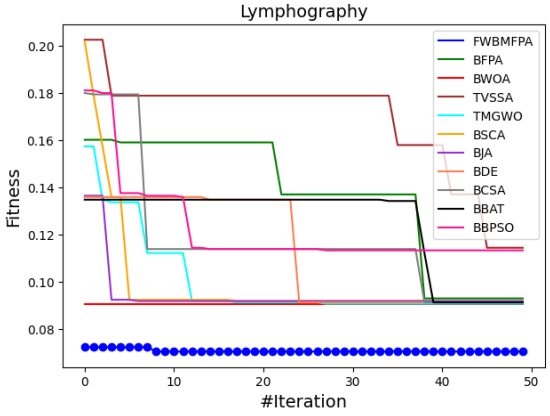

BreastEW
Fitness
#Iteration
FWBMFPA
BFPA
BWOA
TVSSA
TMGWO
BSCA
BJA
BDE
BCSA
BBAT
BBPSO

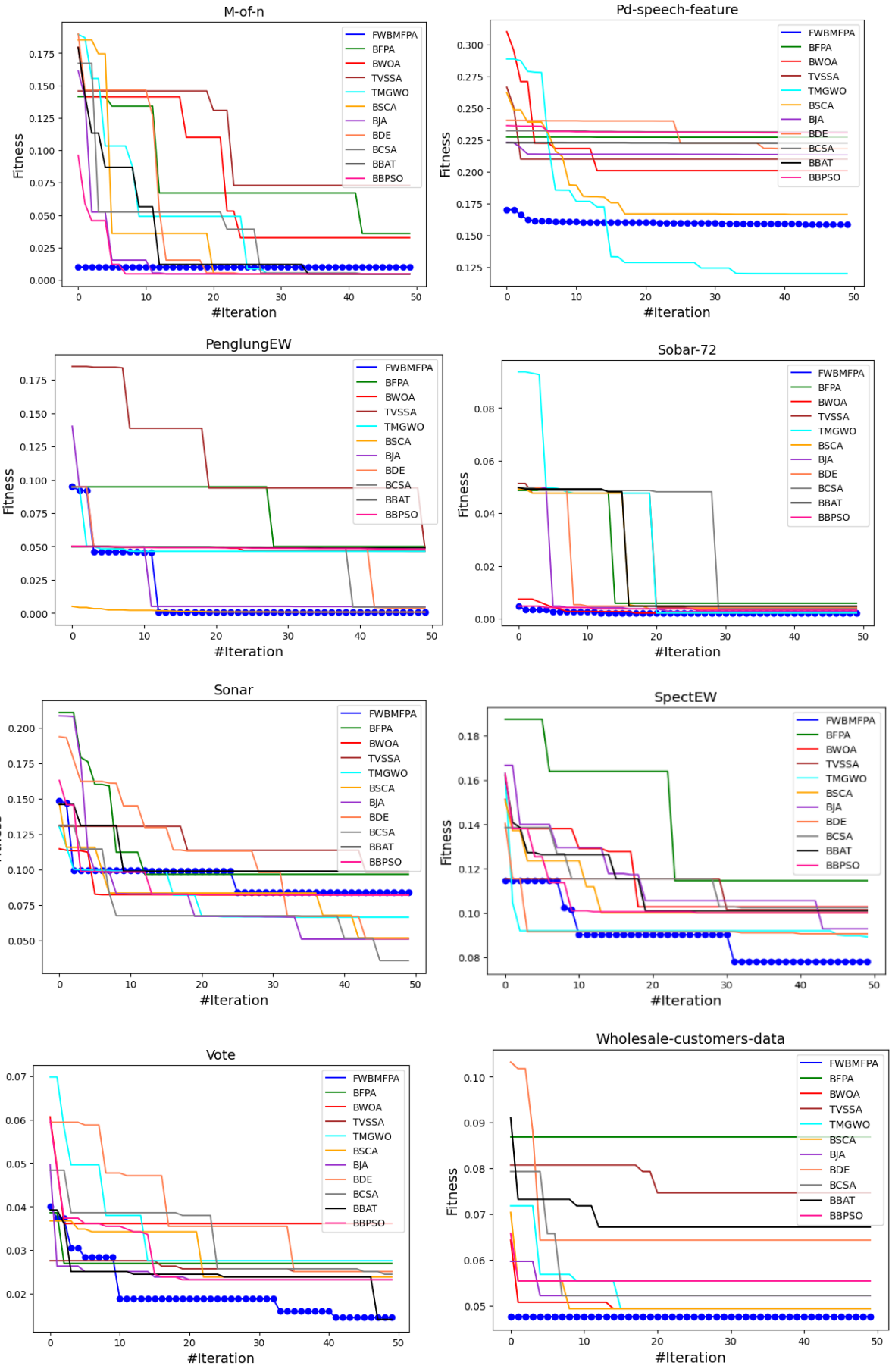

CongressEW
Fitness
#Iteration
FWBMFPA
BFPA
BWOA
TVSSA
TMGWO
BSCA
BJA
BDE
BCSA
BBAT
BBPSO


HeartEW
Fitness
#Iteration
FWBMFPA
BFPA
BWOA
TVSSA
TMGWO
BSCA
BJA
BDE
BCSA
BBAT
BBPSO


Ionosphere
Fitness
#Iteration
FWBMFPA
BFPA
BWOA
TVSSA
TMGWO
BSCA
BJA
BDE
BCSA
BBAT
BBPSO


lung-cancer
Fitness
#Iteration
FWBMFPA
BFPA
BWOA
TVSSA
TMGWO
BSCA
BJA
BDE
BCSA
BBAT
BBPSO


Lymphography
Fitness
#Iteration
FWBMFPA
BFPA
BWOA
TVSSA
TMGWO
BSCA
BJA
BDE
BCSA
BBAT
BBPSO


M-of-n
Pd-speech-feature
PenglungEW
Sobar-72
Sonar
SpectEW
Vote
Wholesale-customers-data
Fitness
#Iteration
FWBMFPA
BFPA
BWOA
TVSSA
TMGWO
BSCA
BJA
BDE
BCSA
BBAT
BBPSO

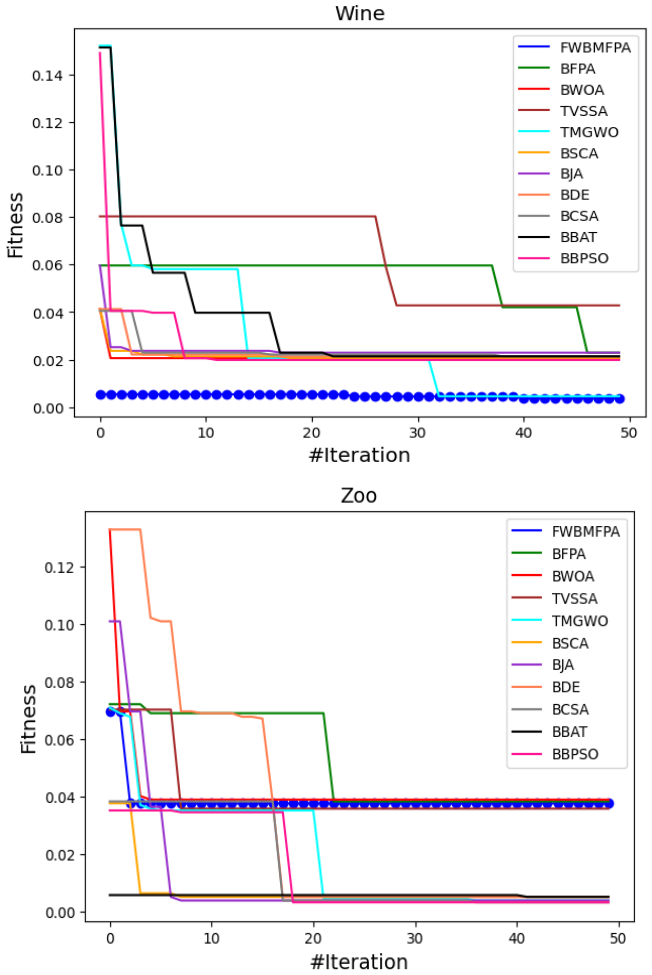


Fig. 3. FWMBFPA fitness values compared to other methods with KNN classifier.

TABLE 8
Classification accuracy obtained by FWMBFPA and other methods

Dataset	**FWBMFPA**	**BFPA**	**BWOA**	**TVSSA**	**TMGWO**	**BSCA**	**BJA**	**BDE**	**BCSA**	**BBAT**	**BBPSO**
Algerian forest fires	1	0.9864	0.9864	1	1	1	1	1	1	1	1
Breast cancer	0.9952	0.9809	0.9714	0.9809	0.9904	0.9857	0.9857	0.9809	0.9857	0.9857	0.9904
BreastEW	0.9707	0.9649	0.9649	0.9649	0.9707	0.9707	0.9649	0.9649	0.9649	0.9649	0.9649
CongressEW	0.9923	0.9694	0.9847	0.9694	0.9770	0.9770	0.9847	0.9874	0.9847	0.9847	0.9847
HeartEW	0.9135	0.8395	0.8395	0.8518	0.8765	0.8641	0.8641	0.8765	0.8765	0.8765	0.8765
Ionosphere	0.9716	0.9150	0.9150	0.9339	0.9622	0.9622	0.9433	0.9433	0.9433	0.9339	0.9622
lung-cancer	1	1	0.9000	1	1	1	1	1	1	1	1
Lymphography	0.9333	0.9111	0.9111	0.8888	0.9111	0.9111	0.9111	0.9111	0.9111	0.9111	0.8888
M-of-n	1	0.9300	0.9733	0.9333	1	1	1	1	1	1	1
Pd-speech-feature	0.8414	0.7753	0.7973	0.8193	0.8810	0.8325	0.7885	0.7841	0.7709	0.7797	0.7709
penglung	1	0.9545	0.9545	0.9545	0.9545	1	1	1	1	0.9545	0.9545
Sobar72	1	1	1	1	1	1	1	1	1	1	1
Sonar	0.9206	0.9047	0.9206	0.9047	0.9206	0.9365	0.9523	0.9523	0.9682	0.9523	0.9206
SpectEW	0.9259	0.8888	0.9012	0.8888	0.9135	0.9012	0.9135	0.9135	0.9012	0.9012	0.9012
Vote	0.9888	0.9777	0.9666	0.9777	0.9777	0.9777	0.9777	0.9777	0.9777	0.9888	0.9777
Wholesale customers data	0.9469	0.9469	0.9545	0.9318	0.9545	0.9545	0.9545	0.9393	0.9545	0.9393	0.9545
Wine	1	0.9814	0.9814	0.9814	1	0.9814	0.9814	0.9814	0.9814	0.9814	0.9814
Zoo	0.9677	0.9677	0.9677	0.9677	1	1	1	1	1	1	1

Based on each method, Table 9. shows the number of features selected. FWMBFPA achieved the lowest number of selected features in all 18 datasets (100%), which is outstanding compared to other methods. Other methods have a percent superiority of less than 17% in the selected feature, and BFPA has selected the fewest features in only one data set, showing how much FWMBFPA has affected BFPA. Therefore, FWMBFPA has superior performance in both classification accuracy and number of selected features, while BFPA has improved its performance. As can be seen in Fig. 4, FWMBFPA selects features with an average accuracy, and other methods select features with an average feature selection. In the diagram, it is apparent that fewer features have been selected more accurately, improving BFPA's performance.

TABLE 9
The number of selected features by FWMBFPA and other methods

Dataset	**FWBMFPA**	**BFPA**	**BWOA**	**TVSSA**	**TMGWO**	**BSCA**	**BJA**	**BDE**	**BCSA**	**BBAT**	**BBPSO**
Algerian forest fires	2	3	3	3	2	3	2	2	3	2	2
Breast cancer	4	4	3	3	4	3	6	3	4	5	4
BreastEW	2	14	5	7	4	4	4	5	9	11	7
CongressEW	4	7	6	5	8	6	6	3	5	4	5
HeartEW	3	5	8	4	7	3	3	4	5	4	3
Ionosphere	3	15	4	12	5	4	15	9	16	12	8
lung-cancer	6	23	9	22	38	7	16	16	20	24	15
Lymphography	5	9	5	7	5	7	7	6	10	6	5
M-of-n	6	10	8	9	6	6	6	6	6	6	6
Pd-speech	14	353	22	74	165	57	307	357	339	337	265
penglung	20	149	48	116	25	25	159	125	147	131	96
Sobar72	3	4	4	5	4	4	5	4	4	4	4
Sonar	9	30	22	25	14	10	29	19	28	27	17
SpectEW	4	10	11	7	5	5	10	11	10	7	5
Vote	2	8	4	4	2	2	2	3	4	5	2
Wholesale customers data	1	4	3	4	3	3	3	3	3	2	5
Wine	2	6	3	3	6	3	4	4	4	4	3
Zoo	5	5	6	6	6	8	6	7	5	6	5

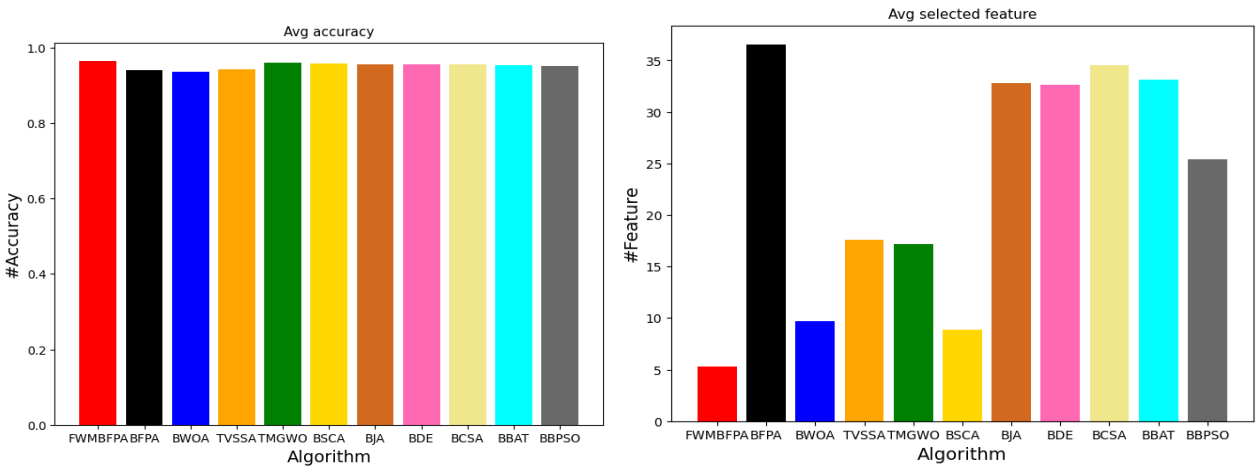


Fig. 4. Average accuracy and average selected features by FWMBFPA and the other methods

5.3.3. Statistical Analysis

In order to rigorously assess the statistical significance of the observed performance differences between the compared feature selection algorithms across the 18 datasets, a non-parametric statistical analysis was carried out. A Friedman test was performed independently on the classification accuracy results (Table 8) and the number of selected features results (Table 9), which are suitable non-parametric alternatives to ANOVA for comparing multiple groups over multiple test conditions. The Friedman test yielded a statistic of 65.0219 with a corresponding p-value of 0. Since the p-value ($p<0.05$), which is less than the significance level $\alpha=0.05$, the null hypothesis is rejected. There is a statistically significant difference in classification accuracy among the algorithms evaluated. For the number of selected features, the Friedman test yielded a statistic of 61.8933 with a p-value of 0. This p-value is also less than $\alpha=0.05$, leading to the rejection of the null hypothesis and confirming a statistically significant difference in the number of features selected by the algorithms.

Since the Friedman test showed significant differences for both evaluation metrics, the Nemenyi post-hoc test was conducted to determine which specific pairs of algorithms performed statistically significantly differently. Nemenyi tests compare the average ranks of algorithms across datasets. In Table 10, the average ranks for classification accuracy as well as the number of selected features are sorted by the average rank for accuracy. The Nemenyi test determines statistically significant differences between algorithms when their average rank differences exceed the Critical Difference (CD). The calculated CD value for comparing 11 algorithms over 18 datasets at a significance level of $\alpha=0.05$ is approximately $CD\approx3.345$.

Table 10. shows that the proposed FWMBFPA achieved the lowest average rank for classification accuracy (3.3889) and the lowest average rank for the number of selected features (1.8889). For classification accuracy, FWMBFPA's average rank (3.3889) is lower than all other algorithms. Comparing FWMBFPA to other algorithms using the CD (3.345):

- FWMBFPA's average rank is significantly lower than algorithms whose average rank is greater than 3.3889+3.345=6.7339. Based on Table 10, FWMBFPA shows a statistically significantly higher accuracy compared to BWOA (8.0556), TVSSA (8.2500), and BFPA (8.6111), as their average ranks are greater than 6.7339.
- For the remaining algorithms (TMGWO, BSCA, BJA, BDE, BCSA, BBAT, BBPSO), the difference in average rank compared to FWMBFPA is less than or equal to the CD, indicating no statistically significant difference in accuracy compared to FWMBFPA at the 0.05 significance level. However, FWMBFPA holds the best average rank among all.

For the number of selected features, FWMBFPA obtained an outstanding average rank of 1.8889. Comparing FWMBFPA's rank to others using the CD (3.345):

- FWMBFPA's average rank (1.8889) is significantly lower than all other algorithms, whose average rank is greater than 1.8889+3.345=5.2339. Based on Table 10, FWMBFPA selects a statistically significantly lower number of features compared to TMGWO (5.3889), BWOA (5.6111), BDE (5.6667), TVSSA (6.7222), BJA (6.8333), BCSA (7.3889), BBAT (6.5000), and BFPA (8.8889).
- The difference in average rank between FWMBFPA (1.8889) and BSCA (4.2222) is $|1.8889-4.2222|=2.3333$, which is less than the CD (3.345).
- The difference in average rank between FWMBFPA (1.8889) and BBPSO (4.4444) is $|1.8889-4.4444|=2.5555$, which is less than the CD (3.345). Therefore, there is no statistically significant difference in the number of selected features between FWMBFPA and BSCA or BBPSO, although FWMBFPA still has the lowest average rank.

Based on the statistical analysis, the proposed FWMBFPA achieved not only the best average rankings for classification accuracy and feature selection but also demonstrated statistically significant superiority in accuracy over several algorithms as well as a statistically significant ability to select fewer features than most of the algorithms evaluated. Statistical evidence supports the effectiveness of the proposed method based on these results.

Fig. 5 and 6 provide a visual representation of these comparisons.

TABLE 10
Average Ranks of Algorithms on 18 Datasets (Lower Rank is Better)

Algorithm	**Avg. Rank (Accuracy)**	**Rank (Accuracy)**	**Avg. Rank (Features)**
FWMBFPA	3.3889	1	1.8889
TMGWO	4.5278	2	5.3889
BSCA	5.0833	3	4.2222
BJA	5.1944	4	6.8333
BCSA	5.3611	5	7.3889
BDE	5.4167	6	5.6667
BBAT	5.8333	7	6.5000
BBPSO	6.2778	8	4.4444
BWOA	8.0556	9	5.6111
TVSSA	8.2500	10	6.7222
BFPA	8.6111	11	8.8889
Critical Difference (CD) for Nemenyi Test ($\alpha=0.05\alpha=0.05$)	≈ 3.345		

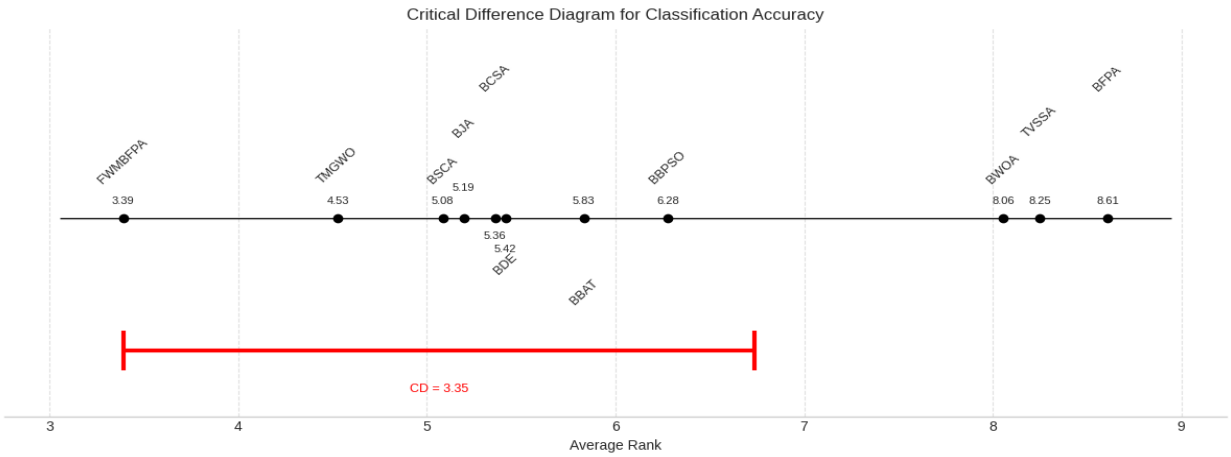


Fig. 5. CD diagram showing the average ranks for Classification Accuracy. The red bar indicates the Nemenyi Critical Difference (α=0.05)

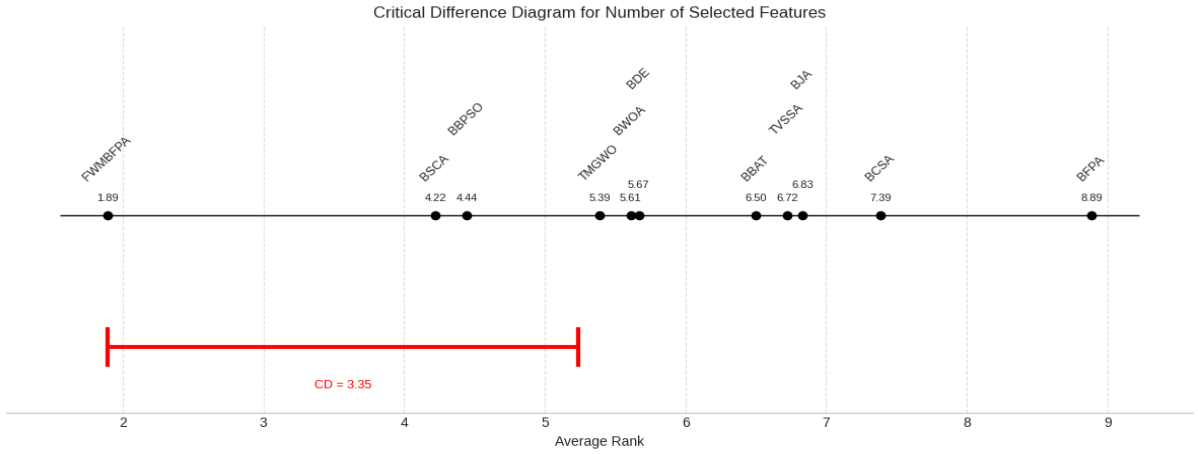


Fig. 6. CD diagram showing the average ranks for the Number of Selected Features. The red bar indicates the Nemenyi Critical Difference (α=0.05)

TABLE 11
Time complexity analysis

Algorithm Type	Representative Algorithms	Theoretical Time Complexity
Wrapper-based Metaheuristics	BFPA, BWOA, TVSSA, TMGWO, BSCA, BJA, BDE, BCSA, BBAT, BBPSO	$O(T\times P\times N\times D)$
Hybrid Filter-Wrapper (Proposed)	FWMBFPA	$O(D^2\times N+T\times P\times N\times D')$

5.3.4. Time Complexity Analysis

This section analyzes the theoretical time complexity of the proposed FWMBFPA and compares it with the general complexity of wrapper-based metaheuristic algorithms. The time complexity of a typical population-based metaheuristic is primarily determined by the number of iterations (T), the population size (P), and the cost of evaluating the fitness function for each solution in every iteration. In feature selection using a classifier like KNN, the fitness evaluation for a subset of F features on N samples has a complexity of approximately $O(N\times F)$. Thus, a standard wrapper-based metaheuristic operating on the original D features has a theoretical complexity of $O(T\times P\times N\times D)$. The proposed FWMBFPA, being a hybrid approach, includes an initial filter phase. This phase involves calculating pairwise Spearman correlations among D features, which takes about $O(D^2\times N)$, and assessing feature relevance, which is $O(D\times N)$. This filter phase is performed only once. The subsequent wrapper phase then operates on the reduced set of D' features. The fitness evaluation in this phase has a complexity of $O(N\times D')$. Therefore, the complexity of the wrapper phase is $O(T\times P\times N\times D')$. The total theoretical complexity of FWMBFPA is the sum of the complexities of the two phases: $O(D^2\times N+T\times P\times N\times D')$.

Table 11. summarizes this comparison. Theoretically, FWMBFPA introduces an initial cost ($O(D^2\times N)$) that is absent in pure wrapper methods. However, for datasets with a large number of original features (D) where the filter phase effectively reduces the feature space to a much

smaller number of features ($D' \ll D$), the cost per iteration in the wrapper phase $O(N \times D')$ becomes significantly lower than $O(N \times D)$. If the total number of fitness evaluations ($T \times P$) is sufficiently large, the cumulative savings in the wrapper phase can potentially outweigh the initial filter cost, making FWMBFPA theoretically more efficient for high-dimensional problems with high redundancy/irrelevance. This aligns with our experimental results showing significant feature reduction on such datasets.

This analysis provides an asymptotic upper bound. Several variables can influence the empirical runtime, such as implementation details, hardware specifications, and constant factors hidden in the Big O notation. For high-dimensional feature selection problems, the theoretical comparison highlights the structural advantage of incorporating a filter phase.

5.3.5. Discussion on Observed Performance and Potential Limitations

The experimental results presented in Tables 4, 8, and 9. show that the proposed FWMBFPA exhibits significant improvements in performance compared to BFPA and several other metaheuristic algorithms, especially in achieving higher classification accuracy with a substantially reduced number of selected features on many datasets. In particular, datasets like 'Pd-speech-feature' and 'penglung' show dramatic feature reductions and significant improvements in accuracy.

This magnitude of improvement, especially when combined with substantial feature reduction and increased accuracy, may seem unusual, raising concerns about dataset bias. The validity of our experimental setup and results has been thoroughly reviewed. For reducing random effects, experiments were run using a standard method, using an 80/20 train/test split and the recommended K-nearest neighbor classifier (K=5).

In FWMBFPA, the observed performance is primarily the result of the synergistic effects of the integrated filter and wrapper phases. Spearman correlation and relevance measures are used in the initial filter phase to eliminate highly redundant and irrelevant features before the search process even begins. Preprocessing reduces the search space and provides the wrapper phase with a more refined, less noisy set of candidate features. Therefore, the Modified Binary Flower Pollination Algorithm in the wrapper phase can better explore and exploit this reduced, relevant feature space to identify a truly optimal or near-optimal subset. As a result of this two-stage approach, FWMBFPA avoids becoming trapped in local optima caused by irrelevant or redundant features and focuses instead on choosing the discriminative subset, which results in both higher accuracy and a significantly smaller feature set on certain datasets, particularly those with very high initial dimensionality and presumably a high percentage of irrelevant/redundant features.

However, it is important to consider potential limitations of FWMBFPA, even though it has shown good performance across the evaluated UCI datasets. The filter phase's effectiveness is determined by the dataset's characteristics, namely its redundancy and irrelevance. In datasets with highly interacting features or less clear-cut redundancy, the initial reduction may be less drastic. The performance of metaheuristic algorithms can also be influenced by parameter tuning and stochasticity. In the future, the method should be evaluated on a wider range of dataset types and its robustness should be enhanced by exploring adaptive thresholds.

6. CONCLUSION AND FUTURE WORK

Dimensionality reduction is essential in many fields because of big data. In this work, a hybrid version of the modified flower pollination algorithm inspired by nature was presented. To reduce the computational overhead and costs associated with the dataset, two filter methods were applied in the first step. In the wrapping step, an optimal set of features has been selected by the modified flower pollination algorithm after redundant and irrelevant features have been discarded. Besides FPA and FWMBFPA, nine other algorithms were evaluated using 18 standard UCI datasets. A KNN classifier was used to learn classification rules. FWMBFPA significantly improved classification accuracy as well as feature selection over FPA. A robust and stable approach is also demonstrated using standard evaluation criteria. FWMBFPA has shown superior performance in terms of accuracy and feature reduction on the evaluated datasets, particularly as a result of the efficient pre-processing by the filter phase. However, the degree of improvement may vary depending on the characteristics of the dataset. To further enhance its robustness and generalizability, this framework will be applied to a wider variety of real-world problems, adaptive filter thresholds explored, and its performance examined on datasets with different feature dependency structures. In the future, parallel processing will speed up the training of classifiers since many feature vectors constitute a computational bottleneck.

Declarations

Ethics approval and consent to participate
Not applicable

Consent for publication
Not applicable

Availability of data and materials
The datasets generated and/or analyzed during the current study are not publicly available due but are available from the corresponding author on reasonable request.

Competing interests
The authors declare that they have no competing interests

Funding
Not applicable

Authors' contributions
Mohammad Ansari Shiri: Programming, software development, Ideas
Najme Mansouri: Testing of existing code components,

writing- original draft preparation

References

[1] P. Dhal and C. Azad. (2023, Sep.). A Lightweight Filter Based Feature Selection Approach for Multi-Label Text Classification. *Journal of Ambient Intelligence and Humanized Computing*. [Online]. *14(9)*, pp. 12345–12357. Available: https://doi.org/10.1007/s12652-022-04335-5

[2] X. Cui, Y. Li, J. Fan, and T. Wang. (2022, May). A Novel Filter Feature Selection Algorithm Based on Relief. *Applied Intelligence*. [Online]. *52(5)*, pp. 5063–5081. Available: https://doi.org/10.1007/s10489-021-02659-x

[3] S. S. Shekhawat, H. Sharma, S. Kumar, A. Nayyar, and B. Qureshi. (2021, Jan.). bSSA: Binary Salp Swarm Algorithm with Hybrid Data Transformation for Feature Selection. *IEEE Access*. [Online]. *9*, pp. 14867–14882. Available: https://doi.org/10.1109/ACCESS.2021.3049547

[4] F. Kılıç, Y. Kaya, and S. Yildirim. (2021, Mar.). A Novel Multi-Population Based Particle Swarm Optimization for Feature Selection. *Knowledge-Based Systems*. [Online]. *219*, p. 106894. Available: https://doi.org/10.1016/j.knosys.2021.106894

[5] M. Kelidari and J. Hamidzadeh. (2021, Apr.). Feature Selection by Using Chaotic Cuckoo Optimization Algorithm with Levy Flight, Opposition-Based Learning and Disruption Operator. *Soft Computing*. [Online]. *25(4)*, pp. 2911–2933. Available: https://doi.org/10.1007/s00500-020-05349-x

[6] D. Wang, H. Chen, T. Li, J. Wan, and Y. Huang. (2020, Dec.). A Novel Quantum Grasshopper Optimization Algorithm for Feature Selection. *International Journal of Approximate Reasoning*. [Online]. *127*, pp. 33–53. Available: https://doi.org/10.1016/j.ijar.2020.08.010

[7] Y. Xue, X. Cai, and W. Jia. (2023, Jun.). Particle Swarm Optimization Based on Filter-Based Population Initialization Method for Feature Selection in Classification. *Journal of Ambient Intelligence and Humanized Computing*. [Online]. *14(6)*, pp. 7355–7366. Available: https://doi.org/10.1007/s12652-022-04444-1

[8] N. S. M. Nafis and S. Awang. (2021, Apr.). An Enhanced Hybrid Feature Selection Technique Using Term Frequency-Inverse Document Frequency and Support Vector Machine-Recursive Feature Elimination for Sentiment Classification. *IEEE Access*. [Online]. *9*, pp. 52177–52192. Available: https://doi.org/10.1109/ACCESS.2021.3069001

[9] L. Peng, Z. Cai, A. A. Heidari, L. Zhang, and H. Chen. (2023, Nov.). Hierarchical Harris Hawks Optimizer for Feature Selection. *Journal of Advanced Research*. [Online]. *53*, pp. 261–278. Available: https://doi.org/10.1016/j.jare.2023.01.014

[10] K. K. Ghosh, S. Ahmed, P. K. Singh, Z. W. Geem, and R. Sarkar. (2020, Apr.). Improved Binary Sailfish Optimizer Based on Adaptive β-Hill Climbing for Feature Selection. *IEEE Access*. [Online]. *8*, pp. 83548–83560. Available: https://doi.org/10.1109/ACCESS.2020.2991543

[11] W. Ma, X. Zhou, H. Zhu, L. Li, and L. Jiao. (2021, Aug.). A Two-Stage Hybrid Ant Colony Optimization for High-Dimensional Feature Selection. *Pattern Recognition*. [Online]. *116*, p. 107933. Available: https://doi.org/10.1016/J.PATCOG.2021.107933

[12] M. Mafarja, I. Aljarah, A. A. Heidari, A. I. Hammouri, H. Faris, A. M. Al-Zoubi, and S. Mirjalili. (2018, Apr.). Evolutionary Population Dynamics and Grasshopper Optimization Approaches for Feature Selection Problems. *Knowledge-Based Systems*. [Online]. *145*, pp. 25–45. Available: https://doi.org/10.1016/j.knosys.2017.12.037

[13] M. M. Mafarja and S. Mirjalili. (2017, Oct.). Hybrid Whale Optimization Algorithm with Simulated Annealing for Feature Selection. *Neurocomputing*. [Online]. *260*, pp. 302–312. Available: https://doi.org/10.1016/j.neucom.2017.04.053

[14] H. Faris, A. A. Heidari, A. M. Al-Zoubi, M. Mafarja, I. Aljarah, M. Eshtay, and S. Mirjalili. (2020, Mar.). Time-Varying Hierarchical Chains of Salps with Random Weight Networks for Feature Selection. *Expert Systems with Applications*. [Online]. *140*, p. 112898. Available: https://doi.org/10.1016/J.PATCOG.2021.107933

[15] A. Faramarzi, M. Heidarinejad, B. Stephens, and S. Mirjalili. (2020, Mar.). Equilibrium Optimizer: A Novel Optimization Algorithm. *Knowledge-Based Systems*. [Online]. *191*, p. 105190. Available: https://doi.org/10.1016/j.knosys.2019.105190

[16] P. Agrawal, H. F. Abutarboush, T. Ganesh, and A. W. Mohamed. (2021, Mar.). Metaheuristic Algorithms on Feature Selection: A Survey of One Decade of Research (2009–2019). *IEEE Access*. [Online]. *9*, pp. 26766–26791. Available: https://doi.org/10.1109/TMTT.2021.3056433

[17] A. M. Anter and M. Ali. (2020, Mar.). Feature Selection Strategy Based on Hybrid Crow Search Optimization Algorithm Integrated with Chaos Theory and Fuzzy C-Means Algorithm for Medical Diagnosis Problems. *Soft Computing*. [Online]. *24(3)*, pp. 1565–1584. Available: https://doi.org/10.1007/s00500-019-03988-3

[18] L. Abualigah, B. Alsalibi, M. Shehab, M. Alshinwan, A. M. Khasawneh, and H. Alabool. (2021, Sep.). A Parallel Hybrid Krill Herd Algorithm for Feature Selection. *International Journal of Machine Learning and Cybernetics*. [Online]. *12*, pp. 783–806. Available: https://doi.org/10.1007/s13042-020-01202-7

[19] M. Alweshah, S. Alkhalaileh, D. Albashish, M. Mafarja, Q. Bsoul, and O. Dorgham. (2021, Jul.). A Hybrid Mine Blast Algorithm for Feature Selection Problems. Soft Computing. [Online]. 25, pp. 517–534. Available: https://doi.org/10.1007/s00500-020-05164-4

[20] E. Nabil. (2016, Feb.). A Modified Flower Pollination Algorithm for Global Optimization. *Expert Systems with Applications*. [Online]. *57*, pp. 192–203. Available: https://doi.org/10.1016/j.eswa.2016.03.005

[21] Z. A. A. Alyasseri, A. T. Khader, M. A. Al-Betar, M. A. Awadallah, and X.-S. Yang. (2018). Variants of the Flower Pollination Algorithm: A Review. In Nature-Inspired Algorithms and Applied Optimization, pp. 91–118. [Online]. Available: https://doi.org/10.1007/978-3-319-67669-2_5

[22] X.-S. Yang. (2012). Flower Pollination Algorithm for Global Optimization. In International Conference on Unconventional Computing and Natural Computation, Lecture Notes in Computer Science, vol. 7445, pp. 240–249. Springer. Available: https://doi.org/10.1007/978-3-642-32894-7_27

[23] M. Abdel-Basset and L. A. Shawky. (2019, Dec.). Flower Pollination Algorithm: A Comprehensive Review. *Artificial Intelligence Review*. [Online]. *52*, pp. 2533–2557. Available: https://doi.org/10.1007/s10462-018-9624-4

[24] M. Abdel-Basset, R. Mohamed, S. Saber, S. S. Askar, and M. Abouhawwash. (2021, Jul.). Modified Flower Pollination Algorithm for Global Optimization. *Mathematics*. [Online]. *9(14)*, p. 1661. Available: https://doi.org/10.3390/math9141661

[25] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili. (2017, Dec.). Salp Swarm Algorithm: A Bio-Inspired Optimizer for Engineering Design Problems. *Advances in Engineering Software*. [Online]. *114*, pp. 163–191. Available: https://doi.org/10.1016/j.advengsoft.2017.07.002

[26] A. E. Hegazy, M. A. Makhlouf, and G. S. El-Tawel. (2019, Apr.). Feature Selection Using Chaotic Salp Swarm Algorithm for Data Classification. *Arabian Journal for Science and Engineering*. [Online]. *44*, pp. 3801–3816. Available: https://doi.org/10.1007/s13369-018-3680-6

[27] A. K. Naik, V. Kuppili, and D. R. Edla. (2020, Jun.). Efficient Feature Selection Using One-Pass Generalized Classifier Neural Network and Binary Bat Algorithm with a Novel Fitness Function. *Soft Computing*. [Online]. *24(6)*, pp. 4575–4587. Available: https://doi.org/10.1007/s00500-019-04218-6

[28] Y. Gao, Y. Zhou, and Q. Luo. (2020, Aug.). An Efficient Binary Equilibrium Optimizer Algorithm for Feature Selection. *IEEE Access*. [Online]. *8*, pp. 140936–140963. Available: https://doi.org/10.1109/ACCESS.2020.3013617

[29] S. Saremi, S. Mirjalili, and A. Lewis. (2017, Mar.). Grasshopper Optimisation Algorithm: Theory and Application. *Advances in Engineering Software*. [Online]. *105*, pp. 30–47. Available: https://doi.org/10.1016/j.advengsoft.2017.01.004

[30] S. Zhao, P. Wang, A. A. Heidari, X. Zhao, C. Ma, and H. Chen. (2022). An Enhanced Cauchy Mutation Grasshopper Optimization with Trigonometric Substitution: Engineering Design and Feature Selection. *Engineering with Computers*. [Online]. *38(Suppl 5)*, pp. 4583–4616. Available: https://doi.org/10.1007/s00366-021-01448-x

[31] B. Doğan and T. Ölmez. (2015, Jan.). A New Metaheuristic for Numerical Function Optimization: Vortex Search Algorithm. *Information Sciences*. [Online]. *293*, pp. 125–145. Available: https://doi.org/10.1016/j.ins.2014.08.053

[32] F. S. Gharehchopogh, I. Maleki, and Z. A. Dizaji. (2022, Mar.). Chaotic Vortex Search Algorithm: Metaheuristic Algorithm for Feature Selection. *Evolutionary Intelligence*. [Online]. *15(3)*, pp. 1777–1808. Available: https://doi.org/10.1007/s12065-021-00590-1

[33] N. Bacanin, K. Venkatachalam, T. Bezdan, M. Zivkovic, and M. Abouhawwash. (2023, Feb.). A Novel Firefly Algorithm Approach for Efficient Feature Selection with COVID-19 Dataset. *Microprocessors and Microsystems*. [Online]. *98*, p. 104778. Available: https://doi.org/10.1016/j.micpro.2023.104778

[34] Y. Liu, X. Zou, S. Ma, M. Avdeev, and S. Shi. (2022, Oct.). Feature Selection Method Reducing Correlations Among Features by Embedding Domain Knowledge. *Acta Materialia*. [Online]. *238*, p. 118195. Available: https://doi.org/10.1016/j.actamat.2022.118195

[35] J. R. Vergara and P. A. Estévez. (2014). A Review of Feature Selection Methods Based on Mutual Information. *Neural Computing and Applications*. [Online]. *24*, pp. 175–186. Available: https://doi.org/10.1007/s00521-013-1368-0

[36] L. Wang, S. Jiang, and S. Jiang. (2021, Nov.). A Feature Selection Method via Analysis of Relevance, Redundancy, and Interaction. *Expert Systems with Applications*. [Online]. *183*, p. 115365. Available: https://doi.org/10.1016/j.eswa.2021.115365

[37] M. Mafarja, A. Qasem, A. A. Heidari, I. Aljarah, H. Faris, and S. Mirjalili. (2020). Efficient Hybrid Nature-Inspired Binary Optimizers for Feature Selection. *Cognitive Computation*. [Online]. *12*, pp. 150–175. Available: https://doi.org/10.1007/s12559-019-09668-6

Journal of Computer and Knowledge Engineering, Vol. 8, No. 2, 2025. (75-88)

Ferdowsi University of Mashhad

Journal of Computer and Knowledge Engineering

https://cke.um.ac.ir

Information and Communication Technology Association of Iran

Influence Maximization in Social Networks Using Discrete Manta-Ray Foraging Optimization Algorithm and Combination of Centrality Criteria*

Research Article

Zaynab Azizpour[1], Saeid Taghavi Afshord[2], Bagher Zarei[3], Mohammad Ali Jabraeil Jamali[4], and Shahin Akbarpour[5]

10.22067/cke.2025.91839.1148

Abstract *Influence Maximization (IM) is a fundamental problem in social network analysis that seeks to identify a small set of highly influential nodes that can maximize the spread of information. Due to its NP-hard nature, finding an exact solution is computationally infeasible for large-scale networks. To address this, this paper introduces an enhanced discrete Manta-Ray Foraging Optimization (MRFO) algorithm tailored for IM. The proposed method integrates degree, closeness, and betweenness centrality measures into the fitness function and introduces a fused centrality index to improve the identification of influential nodes. To handle the discrete search space, the continuous MRFO is adapted with novel discretization mechanisms. Experimental evaluations on five real-world networks (NetScience, Email, Hamsterster, Ego-Facebook, and Pages-PublicFigure) demonstrate that the proposed method achieves higher influence spread compared to existing baseline algorithms, with average improvements of 14.63%, 12.81%, 19.03%, 15.24%, and 18.76%, respectively. These results validate the effectiveness, robustness, and practical applicability of the proposed approach for large-scale IM.*

Key Word Social networks, IM, Manta-Ray Foraging optimization algorithm, Centrality criteria.

1. INTRODUCTION

Complex networks with adjacency matrices are considered as graph G = (V, E), where V and E are edges and vertices of a graph. Each vertex in G demonstrates a user in a social network, and each edge indicates the relation between a pair of users. The size of each network is defined based on the network users, N= |V|, and available links in the network, M =|E|. The network structure is shown as n× n adjacency matrix, A = (aij), where each node can have the values {0, 1}. If user i is connected to user j, then aij =1; otherwise, aij = 0 [1]. Fig1 shows an example of social networks based on the neighbor graph.

As illustrated in Fig. 1, user relationships within a social network are determined based on the network's links and connections. These relationships significantly affect the diffusion of information across the network. Due to the large number of users and the complexity involved in identifying the most influential ones, exhaustive search methods are impractical and computationally expensive.

In recent years, social networks have gained widespread popularity, resulting in an increased impact on various aspects of society. For instance, social networks play a vital role in controlling the spread of diseases, marketing products, and conducting political campaigns such as presidential elections. A fundamental challenge in these contexts is how to effectively select influential users to maximize the spread of information or influence within the network. The IM (IM) problem addresses this challenge by seeking to identify the top k most influential nodes that can generate the largest possible spread of influence throughout the social network [2].

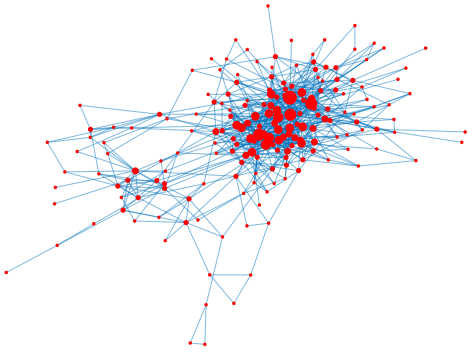


Fig 1. An example of social networks

* Manuscript received 2025 Jenuary 25, Revised 2025 July 9, Accepted 2025 July 12.

[1] Department of Computer Engineering, Shab.C., Islamic Azad University, Shabestar, Iran

[2] Corresponding author. Assistant Professor, Department of Computer Engineering, Shab.C., Islamic Azad University, Shabestar, Iran. **Email:** sa.taghavi@iau.ac.ir

[3] Corresponding author. Assistant Professor, Department of Computer Engineering, Shab.C., Islamic Azad University, Shabestar, Iran. **Email:** zarei.bager@iau.ac.ir

[4] Department of Computer Engineering, Shab.C., Islamic Azad University, Shabestar, Iran

[5] Department of Computer Engineering, Shab.C., Islamic Azad University, Shabestar, Iran

IM is a widely studied topic in the context of social networks. A social network can be modeled as a directed graph, where the users are represented as nodes and their connections as directed edges. Influence spreads throughout the network via the "word-of-mouth" effect, which captures the human-to-human transmission of information or ideas. This process can lead to either a rapid decline or exponential growth in the spread of information.

A key challenge in this domain is to estimate how many users can be influenced by a small group of highly influential individuals. The fundamental goal of IM is to identify an initial set of nodes (seed nodes) that is as small as possible while maximizing the spread of influence across the network. In other words, IM plays a crucial role in viral marketing by helping identify potential customers who can trigger widespread adoption, thereby reducing marketing costs and maximizing profit. Viral marketing leverages word-of-mouth dynamics by targeting a small group of individuals to try a product and encourage broader usage [3].

In practice, IM algorithms determine which nodes should be initially activated. Given a graph G and a parameter K, these algorithms produce an initial seed set by estimating the expected number of nodes that will be influenced through a stochastic diffusion process. The core objective in IM is to maximize the expected size of the final active set while using the smallest possible number of influential users, subject to certain constraints on the initial seed set.

The diffusion process starts with these seed nodes and aims to maximize the overall influence spread within the network. The number of nodes activated during this process determines the effectiveness of the selected seed nodes. IM is an NP-hard problem, meaning that there is no known deterministic polynomial-time algorithm to solve it optimally. Therefore, meta-heuristic optimization methods are commonly employed to find near-optimal subsets of influential users within a reasonable computational time [3,4].

IM is finding a set of influential users, (seed set S) S⊂ V consisting of $K < |V|$ nodes in a social network $G = (V, E)$, Where V is the set of nodes (users), and E is the set of (directed/undirected) edges in G (i.e. social relations between the users). The goal is to maximize K in G through the propagation of the diffusion model. The problem is described by the following equation [3]:

$$IM_M(G, K) = argmax_{e \subseteq V, |e|=k} \sigma_M(e, G) \quad (1)$$

Where σ is a function that calculates the extent of influence for a given set of nodes and represents the spread of influence by activating the set of nodes in e.

Identifying influe ntial users within a network, particularly in large-scale social networks, is a challenging and engaging research problem. Nodes occupy different positions and play various roles within the network, and the effectiveness of influence diffusion largely depends on the underlying network topology. Certain nodes possess structural advantages that make them more effective at spreading information. For instance, central nodes often serve as key conduits for information flow, while nodes with a high number of connections (degree) significantly contribute to influence propagation. Conversely, nodes located at the periphery of the network or those forming isolated clusters may have minimal impact on overall diffusion [5].

Consequently, social importance measures such as degree, betweenness, and closeness centralities — as well as their combined usage — provide valuable insights for identifying the most influential users.

In this study, we propose a discrete method called the Centrality Measure-based Manta-Ray Foraging Optimization (CMMRFO) algorithm, which integrates multiple social importance criteria. The proposed approach has been evaluated using the Facebook dataset. CMMRFO aims to identify a minimal set of users that maximizes influence spread within the network. To achieve this, the algorithm employs a bi-objective fitness function whose weight coefficients are determined by degree, betweenness, closeness centralities, and their combination.

The main contributions of this paper are summarized as follows:

- Development of a discrete version of the MRFO algorithm for solving the discrete IM problem.
- Design of a bi-objective fitness function to simultaneously minimize the number of seed users and maximize overall influence spread.
- Incorporation of social importance measures as weight coefficients in the fitness function.
- Application of degree, betweenness, and closeness centralities, along with their fusion, to guide the search for influential users.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 describes the proposed methodology in detail. Section 4 presents the implementation details and experimental results. Finally, Section 5 concludes the paper and outlines potential future work.

2. BASIC CONCEPTS

2.1. MRFO

The MRFO algorithm is a computational method designed to solve complex optimization problems inspired by the natural foraging behavior of manta rays. In this context, the goal is to determine an optimal path that minimizes the overall cost of travel through multiple locations, analogous to a manta ray visiting several "fish cities." The MRFO algorithm identifies this optimal route through a combination of local search, evolutionary strategies, and similarity-based operations.

In practice, the underlying problem is first formulated as a mathematical optimization task. At each iteration, the algorithm generates a new candidate route for the manta ray. This candidate route is then evaluated and compared with the current best-known route. If the new route yields a lower cost, it replaces the previous best; otherwise, it is discarded. Through iterative refinement, the MRFO algorithm converges toward an optimal or near-optimal solution for the routing problem. This approach can be applied not only to manta ray path planning but also to

other similar route optimization challenges.

1) Mathematical Model: The MRFO algorithm is inspired by three distinct foraging behaviors observed in manta rays: chain foraging, spiral foraging, and storm foraging. These behaviors are mathematically modeled to guide the search process toward optimal solutions, as detailed below.

2) Chain Foraging Strategy: In the chain foraging phase, manta rays detect the location of plankton and swim toward areas with higher concentrations. In the optimization analogy, these high-concentration zones correspond to promising candidate solutions. Although the global optimum is unknown, MRFO assumes that the best solution discovered so far represents the most desirable "plankton" location.

Manta rays are conceptually arranged in a head-to-tail sequence, forming a chain. Except for the leading individual, each manta ray updates its position by moving not only toward the detected food source but also relative to the preceding individual in the chain. This ensures collective information sharing and improved exploration of the search space. At every iteration, each individual's position is refined based on the best solution found up to that point. The mathematical formulation of the chain foraging behavior is provided below.

$$x_i^d(t+1) = \begin{cases} x_i^d(t) + r.\left(x_{best}^d(t) - x_i^d(t)\right) + \alpha.\left(x_{best}^d(t) - x_i^d(t)\right) & i = 1 \\ x_i^d(t) + r.\left(x_{i-1}^d(t) - x_i^d(t)\right) + \alpha.\left(x_{best}^d(t) - x_i^d(t)\right) & i = 2, \dots, N \end{cases} \quad (2)$$

$$\alpha = 2.r.\sqrt{|\log(r)|} \quad (3)$$

Where $x_i^d(t)$ is the position of the ith individual at time t in the dth dimension, r is a random vector in the range [0, 1], while α is weight factor, $x_{best}^d(t)$ is a location with the highest plankton concentrations. Fig 2. displays the behavior of food Search in two-dimensional space. The position Update of the ith individual is determined by the position $x_{i-1}(t)$ for (i-1) the individual and the position $x_{best}(t)$ of the food.

3) Storm Search Strategy: When a group of manta rays detect clusters of plankton in deeper waters, they form a long foraging chain and swim toward the food source using a spiral motion. This spiral foraging strategy resembles the approach used in the Whale Optimization Algorithm (WOA); however, in the MRFO framework, the spiral movement is incorporated specifically in the storm foraging phase. In this strategy, each manta ray moves in a spiral path toward the food source while simultaneously following the preceding individual in the chain. In this way, the manta rays align sequentially and execute a coordinated spiral search to locate and capture plankton more efficiently.

Fig 3. illustrates the storm foraging behavior in a two-dimensional space. In this phase, each individual not only follows the food targeted by his neighbor but also advances toward the food source by executing a spiral trajectory. The mathematical equations that describe this spiral motion in two-dimensional space are defined as follows:

$$\begin{cases} X_i(t+1) = X_{best} + r.\left(X_{i-1}(t) - X_i(t)\right) + e^{b\omega}.cos(2\pi\omega).\left(X_{best} - X_i(t)\right) \\ Y_i(t+1) = Y_{best} + r.\left(Y_{i-1}(t) - Y_i(t)\right) + e^{b\omega}.sin(2\pi\omega).\left(Y_{best} - Y_i(t)\right) \end{cases} \quad (4)$$

Where ω is a random number in [0, 1].

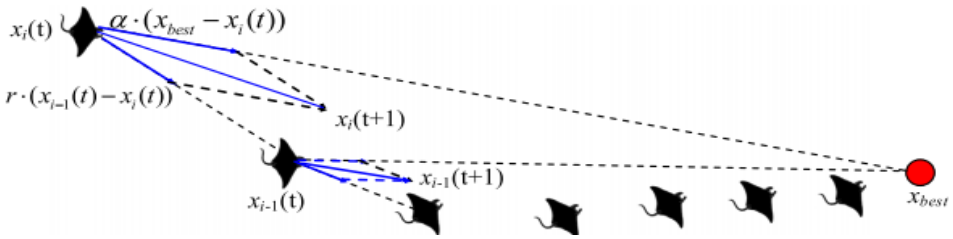


Fig. 2. The behavior of food search in two-dimensional space.

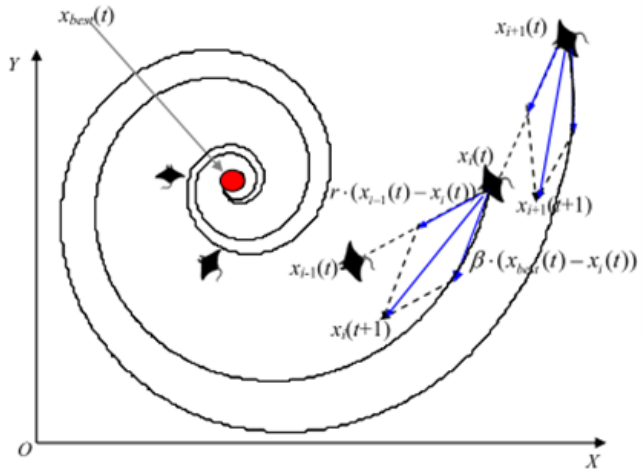


Fig 3. The behavior of storm search in two-dimensional space

This motion behavior may be extended to n-D space. For simplicity, this mathematical model of silicon search can be defined as:

$$parentx_i^d(t+1) =$$

$$\begin{cases} x_{best}^d + r.\left(x_{best}^d(t) - x_i^d(t)\right) + \beta.\left(x_{best}^d(t) - x_i^d(t)\right) & i = 1 \\ x_{best}^d + r.\left(x_{i-1}^d(t) - x_i^d(t)\right) + \beta.\left(x_{best}^d(t) - x_i^d(t)\right) & i = 2, \dots, N \end{cases}$$

$$\beta = 2e^{r_1\frac{T-t+1}{T}}.sin(2\pi r_1)$$

Where β is the weight coefficient, T is the maximum number of iterations, and $r1$ is the random number in [0, 1].

In the cyclone foraging phase, all individuals perform a randomized search using the current best-known food location as their reference point. This mechanism enhances the algorithm's exploitation capability within regions that contain promising solutions. Additionally, this strategy significantly improves the overall search process by enabling individuals to explore new regions. Specifically, each individual can be directed to search for alternative positions that deviate from the current best solution, or it can adopt a completely random position anywhere within the entire search space as a new reference. This balance between local exploitation and global exploration ensures that the MRFO algorithm maintains strong heuristic capabilities while avoiding premature convergence. The corresponding mathematical formulation for this mechanism is provided below.

$$X_{rand}^d = Lb^d + r.(Ub^d - Lb^d)$$

$$x_i^d(t+1) = \begin{cases} x_{rand}^d + r.\left(x_{rand}^d - x_i^d(t)\right) + \beta.\left(x_{rand}^d - x_i^d(t)\right) & i = 1 \\ x_{rand}^d + r.\left(x_{i-1}^d(t) - x_i^d(t)\right) + \beta.\left(x_{rand}^d(t) - x_i^d(t)\right) & i = 2, \dots, N \end{cases}$$

Where x_{rand}^d is a random position randomly produced in a search space. Lb^d and $Ub^d$are the lower and high limits of the search space.

4) Somersault search strategy: In this phase, the location of the food source is treated as a central axis. Each swim around this pivot point and performs somersault-like movements to discover new positions in its vicinity. Through this behavior, individuals continuously update their positions around the best solutions identified so far, enhancing local exploitation while maintaining diversity. The corresponding mathematical formulation for this behavior is presented below.

$$x_i^d(t+1) = x_i^d(t) + S.\left(r_2.x_{best}^d - r_3.x_i^d(t)\right),$$
$$i = 1, \dots, N$$

Where S is the somersault factor that determines the range of movement of Manta Rays movement and S = 2, $r2$ and $r3$ are two random values in the range [0, 1].

As shown in Equation (8), by defining the somersault range, each individual can explore a new search area bounded between its current position and its symmetric counterpart relative to the best position identified thus far. As the distance between an individual's current position and the best-known position decreases, the degree of disturbance applied to its position also diminishes. Consequently, all individuals progressively converge toward the optimal solution within the search space. Therefore, as the number of iterations increases, the somersault foraging range adaptively contracts. Fig 4. illustrates a schematic representation of the somersault foraging behavior in the MRFO algorithm.

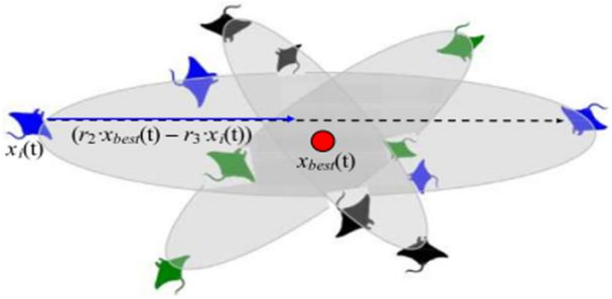


Fig. 4. Somersault foraging behavior in MRFO

Fig 5. illustrates the evolution of three individuals over 100 iterations within the search space based on the corresponding equation. The sampled points are randomly generated between each individual's current position and its symmetric position relative to . As the distance to $xbest$ decreases, the sampled points become more concentrated. This pattern ensures that densely clustered points near $xbest$enhance local exploitation, while more widely distributed points support broader exploration of the search space.

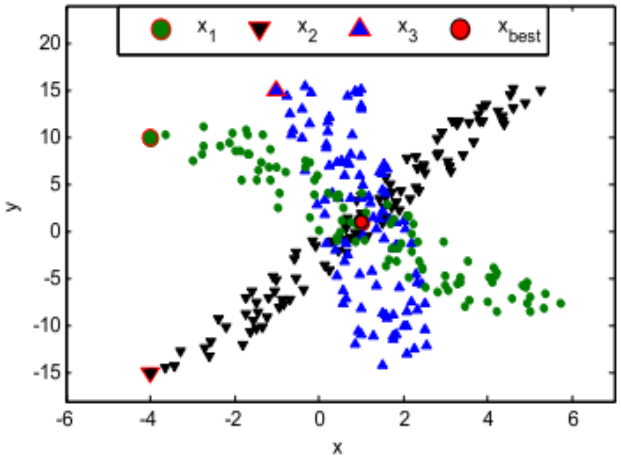


Fig. 5. The Somersault foraging behavior of three individuals in two-dimensional space

Unlike other meta-heuristic optimization methods, MRFO starts the problem by creating the initial population. Each updates their situation by considering the reference and opposite situation. The t/T value is decreased from 1/T to 1 so that heuristic search and application are performed, respectively. The best current solution is selected as a reference situation wage when t/T < U (0,1). A position is randomly created in search space, and It is

selected as the reference position for the heuristic when t/T > U (0,1). It can be moved between chain search behavior and storm search behavior. Subsequently, each updates its position relative to the best position identified through the somersault search mechanism. These updates and computations continue iteratively until the predefined stopping criterion is satisfied. Ultimately, the position and fitness value of the best-performing individual are returned [25].

In the next stage, the proposed method is described in detail. This method is based on the CMMRFO algorithm, an evolutionary approach designed to solve the IM problem. In this approach, measures of user importance within the network are incorporated as key factors. The CMMRFO algorithm enhances solutions through random variations and mutations within the population, iteratively seeking better candidates. By integrating the proposed CMMRFO method with user importance metrics, the influence spread in social networks is effectively maximized, providing a more optimal solution to the IM problem.

3. LITERATURE REVIEW

In this section, the research on IM in social networks is discussed. It has more commercial applications, and so it has been more studied. One of the essential tools for identifying the influenced users is using social importance criteria. These criteria involve degree centrality, betweenness centrality, closeness centrality, and other similar criteria. The users with the maximum network influence are identified using these criteria. In the following, IM methods based on social importance criteria are studied in detail. An effective influence evaluation model based on whole valuation and variance of neighbor nodes valuation has been presented to create unreliable communication channels [6]. Then, the Moth-flame optimization algorithm has been developed to search the set of influence-maximizing nodes by using local intersection and mutation evolution above updating the conventional solution. The criterion of degree centrality was introduced by [7] for the first time and later it was used for IM. Then, using an analysis framework based on modular functions such as BC [8] was shown that a greedy natural strategy obtains a solution that can be proved to be 63% optimum for several class models. This framework suggests a reasoning approach to guarantee the performance of algorithms for this kind of influence problem in social networks. An approach based on PageRank for influence maximizing in a network to search the web has been proposed in [9]. A different approach based on simulated Annealing for IM has been suggested in [10]. This is the first SA-based algorithm for solving this problem. In addition, two heuristic methods have been proposed to accelerate the process of SA convergence, and a new method has been suggested for computation influence to accelerate this algorithm.

Some changes to the IM problem structure were presented by [11] to adjust it with particle swarm intelligent algorithms and to reach a slope in the space of the objective function. The proposed approach was tested using real and artificial data sets. The gray wolf optimization (GWO) algorithm has been considered as a particle swarm intelligence algorithm, along with page ranking and greedy algorithms were used as evaluation methods. The reason for the low performance of greedy approaches was analyzed in [12] and an efficient algorithm called degree-descending search strategy evolution (DDSE) has been proposed. Firstly, a degree-descending search strategy is suggested, which can produce a set of nodes whose influence spread can be compared to the centrality degree. An evolutionary algorithm based on DDSE has been developed that considerably improves efficiency by removing time-consuming simulations of greedy algorithms.

An improved discrete particle swarm optimization algorithm along with an advanced network topology-based strategy for influence maximizing has been proposed in [13]. In this strategy, at first, k-influenced nodes of a temporary optimal seed set are combined in an ascending order based on degree metric so that the nodes with lower degree centrality can utilize preferably the influenced neighbors. In the second step, a local greedy strategy is applied to replace the current node with the most influenced node of each node's direct neighbor node-set of temporary seed set.

An improved greedy-based strategy called Cost-Effective Lazy Forwarding (CELF) has been created [14]. It reduces computation costs twice without damaging precision by utilizing the submodularity of objective function. Later, an optimized version called CELF++ was suggested [13], and the results showed 50% more efficiency improvement compared to CELF.

Time-sensitive centrality criteria were presented for IM in social networks by considering the diffusion value, and direct and indirect neighbors [15]. Hence, four time-sensitive centrality measures, including time-sensitive closeness centrality, time-sensitive harmonic, time-sensitive decay centrality, and time-sensitive eccentricity centrality were proposed.

Degree centrality based on various environments is used by [16] to increase its local search power. Through performing experiments, it has been specified that local search strategies based on different environments have considerable differences in improving the global search of the algorithm, and increasing the DPSO algorithm based on the degree centrality of different environments has considerable influences. Finally, the DPSO-NDC algorithm has been suggested based on the degree of centrality of the best environment with improved local search capability.

A mechanism to measure the influence index in popular social media platforms like Facebook, Twitter, and Instagram was suggested in[17]. Some sets of features determining influence on the consumers are modeled by regression approach. Infrastructure machine learning algorithms involve Ordinary Least Squares (OLS), Regression Nearest Neighbor (KNN), Support Vector Regression (SVR), and Lasso Regression models to compute cumulative scores adopted in terms of influence index. The findings show that Participation,

meta-learning, and feedings are crucial in determining influencers.

An improved discrete differential evolution algorithm (IDDE) based on the network analysis has been suggested in [18]. This algorithm improves the variance of the differential evolution algorithm. After removing the objective node as an index, it gives a discrete number and discrete precision of the remaining network to evaluate the importance of the node and as a result, it presents a health function based on the network power. This method shows symmetry in two aspects. Firstly, when the number of removed objective nodes increases in a social network, global coherence decreases between the network nodes. Secondly, the range of global influence becomes small when the proposed method displays the number of objective nodes. Comparable experiments have been performed on four real-world data sets with different sizes. The results show that the IDDE algorithm outperforms the comparison algorithms.

The authors of [19] present a framework involving community detection in a social network using the Shuffled Frog Leaping algorithm (SFL). This framework aims to maximize influence spread in an independent cascade model. In this framework, various communities are identified in a social network using a community detection algorithm. Then, the SFL algorithm is applied to maximize the influence spread in these communities. The SFL algorithm is an evolutionary algorithm, searching the solutions improvement based on random changes and mutations created in the frog population. Local search strategies, including hill climbing based on lake adoption and user centrality weight, are also used to more improvement in the solutions. These strategies find the best points in the search space by using the weight of the user's centrality, and they make more improvements in the solutions by local search. Therefore, this framework, including community detection, SFL algorithm, and local search strategies, maximizes influence under the independent cascade model, and optimal solutions are provided for this problem.

A meta-heuristic approach based on multi-criteria decision-making (MCDM) has been proposed by the authors of [20] to solve the IM problem in social networks. The MCDM approach selects candidate nodes by removing less-influence nodes in the preliminary step based on the centrality criterion, and it decreases the computational cost. Afterward, an improved version of simulated Annealing (SA) with an advanced search strategy has been suggested to find an optimal solution.

An evolutionary Discrete Crow Search Algorithm (DCSA) using crow swarm intelligence has been suggested by [21] to solve effectively the IM problem. DCSA makes a new coding mechanism and discrete evolution rules. Initialization methods based on degree centrality and random walking strategy are applied to increase searchability. An IM algorithm called Weighted Artificial Bee Colony (WABC) has been proposed by [22]. It is based on a technique inspired by biology to detect the subset of users that maximizes diffusion. WABC has used ranking techniques based on classic centrality criteria. A new approach with multi-feature IM has been suggested by [23]. This approach uses the multi-feature nature of network nodes (age, gender, etc.) to consider specified groups of users. Also, it uses centrality criteria to rank the user's importance in various groups. The Discrete Bat Algorithm (DBA) has been presented by [24]. It is based on partitioning a network and increases the stability of the initial DBA. The experimental results showed that the DBA converges in each run to a specified Local Influence Estimation (LIE) value. It removes the high oscillation phenomenon of the LIE fitness value created by the main DBA. This method has been used centrality criteria for local search in the fitness function. In [29], a novel Multi-objective Cuckoo Search Algorithm (MOCSA) designed for community detection in social networks, emphasizes improved accuracy and efficiency by incorporating a strategy based on close neighbors in the objective function. In [30], a hybrid multi-objective algorithm is presented incorporating multiple optimization techniques and fuzzy clustering that outperforms existing methods in detecting overlapping communities in complex social networks. In [31], the LCD-SN algorithm enabled highly accurate and efficient community detection in social networks by leveraging local node characteristics and neighbor information without dependence on initial seed nodes. In [32], Opinion Leader Selection (OLS) as an optimization problem using bio-inspired algorithms has been formulated. It combined the African Vultures Optimization Algorithm (AVOA) and Hunger Games Search (HGS) for improved leader identification. In [33], the proposed method effectively identified influential opinion leaders in social networks using hybrid optimization algorithms and topological network analysis, achieving higher accuracy and marketing impact than existing approaches. A comparison of the previous works is shown in Table 1.

Despite extensive efforts in IM, existing methods often rely on either single centrality measures or heuristic meta-heuristics that are not fully adapted to the discrete nature of social networks. Many approaches use basic node rankings (e.g., degree or PageRank) without integrating multiple structural properties, which limits their accuracy in identifying truly influential nodes. Additionally, continuous meta-heuristic algorithms are frequently applied with minimal modification, resulting in suboptimal performance when handling the inherently discrete selection of seed nodes. Moreover, some algorithms suffer from high computational costs or slow convergence, especially on large-scale networks. Therefore, there is a clear need for a method that effectively fuses multiple centrality measures within a robust, discrete meta-heuristic framework to achieve higher influence spread while maintaining computational efficiency.

TABLE 1

The previous work's comparison

Ref	Algorithm / Method	Key Idea	Strength	Limitation
[6]	Moth-Flame Optimization	Uses whole valuation & neighbor variance to handle unreliable channels	Handles uncertainty	Limited to specific network conditions
[7][8]	Degree Centrality & Greedy	Early use of centrality; submodular function framework	Theoretical performance guarantee (63% optimal)	Greedy: high time cost, limited scalability
[9]	PageRank	PageRank-based node ranking	Intuitive for web networks	Less effective for general social graphs
[10]	Simulated Annealing	First SA-based IM with acceleration heuristics	Good exploration ability	Slow convergence in large graphs
[11]	Particle Swarm Optimization & GWO	PSO adapted to IM; GWO used for evaluation	Intelligent swarm behavior	May stagnate; lacks centrality integration
[12]	DDSE (Degree-Descending Search Evolution)	Evolutionary search avoiding costly simulation	Faster than greedy	Needs careful parameter tuning
[13]	Improved Discrete PSO	Advanced topology strategy with local greedy replacement	Good local refinement	May get stuck in local optima
[14]	CELF / CELF++	Optimized greedy selection with lazy evaluation	50% faster than CELF	Still costly for large networks
[15]	Time-sensitive Centrality	Four new time-aware centrality measures	Considers diffusion time	High computation for dynamic networks
[16]	DPSO-NDC	Local search in different environments	Improved local/global balance	Sensitive to environment selection
[17]	ML-based Influence Index	Regression models for social media	Leverages user behavior features	Not directly IM for seeding
[18]	Improved Discrete Differential Evolution	Variance-based node ranking; health function	Symmetric handling of removed nodes	Limited to certain network structures
[19]	SFL + Community Detection	Shuffled Frog Leaping with Community Detection	Uses local community structure	Relies on quality of community detection
[20]	MCDM + Improved SA	Node filtering + advanced SA	Reduces cost via pre-selection	SA still has convergence limits
[21]	Discrete Crow Search	Crow swarm intelligence; random walking	Novel coding; better exploration	Lacks robust local refinement
[22]	Weighted Artificial Bee Colony	Biological swarm inspired; uses ranking	Effective for classic criteria	Ranking alone may overlook structural synergy
[23]	Multi-Feature IM	Uses user attributes + centrality	More realistic user modeling	Needs rich attribute data
[24]	Discrete Bat Algorithm	Uses partitioning to stabilize LIE	Better stability; local search	High oscillation is removed but may converge slowly
[29]	Multi-objective Cuckoo Search Algorithm (MOCSA)	Uses cuckoo search with a neighbor-based strategy for accurate community detection.	High detection accuracy and efficiency.	May face convergence issues in very large-scale networks.
[30]	Hybrid Multi-objective Algorithm with Fuzzy Clustering	Integrates multiple optimization techniques and fuzzy clustering for overlapping community detection.	Handles overlapping communities effectively, with better performance than traditional methods.	Increased computational complexity due to the hybrid structure.
[31]	LCD-SN Algorithm	Leverages local node characteristics and neighbor data for community detection without relying on seed nodes.	High accuracy and efficiency, seed-free approach.	May require fine-tuning for networks with sparse connections.
[32]	Opinion Leader Selection (OLS) with AVOA and HGS	Formulates leader selection as an optimization problem using bio-inspired algorithms (AVOA + HGS).	More precise leader identification and robust search capability.	Algorithm performance is sensitive to parameter settings.
[33]	Hybrid Optimization for Opinion Leader Detection	Combines hybrid optimization algorithms with network topology analysis for better opinion leader identification.	Higher accuracy and marketing influence than other methods.	May need high computational resources for very dense networks.

4. THE PROPOSED METHOD

This paper presents a new method called CMMRFO for IM in social networks using the Manta Ray algorithm and a combination of centrality criteria. A weighted combination of criteria, including degree, betweenness, and closeness centralities, are used to compute the social importance of the users. Since the MRFO is continuous, it cannot be used to solve the discrete IM problem. Hence, a discrete method is proposed for the MRFO, and its discrete version is used to solve the IM problem. The details are explained in the following.

4.1. discretization of the MRFO

In meta-heuristic algorithms, discretization converts continuous values into discrete ones. The purpose is to convert continuous search space to discrete search space so that meta-heuristic algorithms can find the problem's best solution. Continuous values are converted like real numbers in search space, and the initial population as discrete values or a set of integers in most meta-heuristic algorithms for discretization. This conversion can be performed as a simple discretization, for instance, by estimating the value to the nearest discrete, or it can be performed as a complex discretization by conversion function or other methods. This conversion to discrete space helps the algorithms search for the best solution in discrete space, and in this way, efficiency improvement and the efficiency of algorithms can be increased. Since the most fundamental problems are not continuous, discretization helps the algorithms reach the optimal value of the problems [26]. In this paper, since the search space is selected among social network users, the initial population's values indicate the user's index in a social network. Hence, this problem does not involve containing values for the initial population. In addition, the severe population cannot include continuous values in the heuristic process in the meta-heuristic fragging optimization algorithm. As a result, the initial population is a limited range of user indexes in the proposed method, which is valid for the centrality threshold. They can be converted to an influence user. An integer for a random variable can be used in the heuristic step in the MRFO instead of X_{rand}^{d} which is a real random variable.

4.2. Computing the social importance of users

Increasing the users and the data volume of these users in social networks requires analyzing and extracting useful information from data. Such information can be useful for different applications like advertisements, marketing procedures of user behavior, etc. In this regard, it is one of the valuable tools in user importance criteria in social networks. These criteria investigate the user's importance in various aspects of social networks, and they are introduced as an effective tool for analyzing social network data. This study uses three criteria of centrality importance involving closeness, betweenness, and a combined criterion of the network. They are explained in detail in the following.

1) Centrality criteria: Centrality criteria are the network analysis measures used to detect the most powerful nodes. The centrality criterion quantifies direct friendship relations for a node in social networks. According to the centrality criterion, the importance of a node is determined based on its degree. Suppose G = (V, E) is a social network. V is a set of n nodes, while (n=|V|) shows the users in a network. E indicates a set of m edges between the users. (m=|E|), shows the relations between the users. Social networks are shown by a symmetric matrix A called adjacency matrix with dimensions n × n. Each entry, a_{ij} is considered the relationship between node i and node j if equal to. According to the centrality criterion, it can be computed for each user as equation 10 [27].

$$C(i) = \sum_{i=1}^{n} \sum_{j=1}^{n-1} \frac{a_{ij}}{n-1} \tag{10}$$

An ij is the relation between the user i and j in the adjacency matrix, and n is the whole number of users in a network. In this case, the value of the degree of centrality can be obtained for all users in a network, and it indicates a favor among social networks. The users having relations with many users have a high value of degree centrality. Such users are exposed to information or data diffusion in social networks. In contrast, such users with a low degree of centrality do not have so much popularity and show introverted personalities. This criterion limitation is local access to the network topology, and it uses limited local knowledge to decide about the user's importance.

2) Closeness criterion: The closeness criterion hypothesizes that the power of an individual has a reverse relation with another individual in that social network in terms of closeness with distance sum [27]. In other words, the closeness criterion for the user is obtained based on whole routes from other users in a social network. The closeness criterion is computed for each user as equation (11) [27].

$$CN(i) = \frac{1}{\sum_{i=1}^{n} \sum_{j=1}^{n-1} d_{ij}} \tag{11}$$

Where d_{ij} Shows the shortest route from node i to node j. The closeness criterion of the user shows the average distance of the user with another user in a social network as a quantitative value. The users with a high closeness criterion receive the information from each point of the network in less expected time because they are less distant from other social network users. In contrast, the user having less closeness criterion and being away from another network user receives data diffusion in networks later than expected. The limitation of this criterion is that disconnected networks do not work well, and they cover only a part of the connected network.

3) Betweenness criterion: According to betweenness criteria, the importance of a user in social networks is defined as the shortest path between other users passing through that point. In other words, the betweenness criteria for each user i is according to the shortest routes between all network users, and user i is located between them. The betweenness criterion is computed for each user as equation (12) [27].

$$B(i) = \sum_{i=1}^{n} \sum_{j=1}^{n-1} \sum_{k=1}^{n-2} \frac{\sigma_{jk}(i)}{\sigma_{jk}} \tag{12}$$

Where σ_{jk}(i) is the shortest route from node j to k, which passes from node i. σ_{jk} shows the number of shortest routes from node j to k. In betweenness criterion data is transferred in a social network between users through the shortest routes. The individual having a high value of betweenness criteria has more control over the information in the whole network. Therefore, it can be a good alternative for selection as a user with high influence in social networks.

4) The combination criterion: Social importance criteria are those used to measure and analyze the user's importance and their communications in social networks. These criteria help us determine which entities have the maximum influence on a network and how communication is made between the users. Social importance criteria point to the influence of a user in a network. The conventional social importance criteria involve the degree of centrality. Degree Centrality measures the number of user communications. Betweenness centrality shows how a user is located in the routes between two other users. Closeness Centrality shows how a user is close to the other users in a network. The Fusion Centrality combines the effect of these three mentioned criteria simultaneously. It is defined as follows.

$$FC = w_1 * DC + w_2 * CC + w_3 * BC,\ w_1, w_2, w_3 \in [0,1] \quad (13)$$

Where FC indicates the combinations of centrality criteria, DC and w1 show the degree centrality criterion and the weight of the degree centrality criterion, respectively. Also, CC and w2 are the closeness centrality criterion and the weight of degree centrality criterion, respectively, BC and w3 refer to betweenness centrality criteria. The weight of each centrality criterion is a value between zero and one.

The classic importance criterion of centrality is determined based on the user's output and input links in a social network. The closeness criterion is specified according to the distance or required steps to reach another user in a network. At last, the combination criterion is made of these three essential criteria and combines previous criteria. To select the influenced users, the combination criterion considers three criteria, degree centrality, closeness centrality, and betweenness centrality. Other criteria investigate the user's importance in a social network in just one dimension. Using these combination criteria to analyze social network data helps users and managers make decisions about using social network data. This paper can be used as a helpful resource to study the combination of importance criteria in social networks and data analysis methods of a social network.

4.3. Solving IM

IM in social networks is detecting a set of individuals in a network with the highest potential to affect others. It is an important problem in the analysis of social networks because it can be used to spread the behaviors or particular ideas in a network or to prevent the spread of negative ideas or behaviors. Meta-heuristic algorithms are optimization algorithms used to solve classic optimization techniques. Meta-heuristic algorithms can be used to detect individuals with the highest potential to affect others in terms of IM in social networks. These algorithms are performed by producing and evaluating potential solutions and then using the best solutions to produce new ones.

1) Coding the initial population: In optimization problems, solved by meta-heuristic methods, initial population planning is an important issue. The purpose of this problem is to consider each member in the initial population as a final solution. Each member can be evaluated based on fitness function to specify the optimal solution. The MRFO is used to find k users among M users having the maximum influence in a social network. The initial population is defined as a vector of Manta rays, and elements connect them, and they constitute a solution. The initial population is defined as a vector of discrete values in this algorithm, and each entry indicates the user in the social network index. The length of this vector is equal to M (the number of users in a search space), and the value of each element shows the users' index in the social network. The value of zero in each element is related to the lack of selection of the related user, while the value of 1 shows the selection of that user. The problem space is limited according to the spread of the social network and the number of users. In other words, the users whose communication exceeds the threshold value are considered the available alternative in the problem space. The threshold value is the mean of the degree centrality for all users in the social network. So, influenced users can be selected among the users whose connection and communication are more than the mean of communication degree in the whole network. Fig 6 shows a sample of the initial solution.

	U1	U2	U3	U4	…	UM
X	5	16	0	8	…	27

Fig. 6. A sample of the initial solution in the proposed method.

According to Fig 6, it can be found that initial populations involve the element equal to the number of users in a search space, where the value of element 0 shows the selection of that user as an influenced user. If the value is non-zero, it shows the user's index in the social network. The users selected as the initial population are considered as the entry of the fitness function trying to compute the influence of users selected as influenced users.

2) Fitness function: The proposed fitness function finds the minimum number of users with maximum influence in social networks. In other words, social importance criteria are determined as an influence parameter factor in the fitness function. The fitness function has two parameters. The first parameter is related to the influence, while the second is related to the number of users in the social network. The proposed fitness function is as follows.

$$max\ f = \sum_{i=1}^{M} W_i SP_i - \sum_{i=1}^{n} W_i N_i \quad s.t.$$
$$\sum_{i=1}^{n} W_i \leq 1 \qquad \sum_{i=1}^{n} N_i \geq 1 \qquad i > 0 \quad (9)$$

Where i shows the user index in the social network in equation 9; M refers to the number of users in the influenced search space; SP_i demonstrates the number of users under the influence; Ni and Wi show the number of influences and the weight of social importance, respectively. Fig 7. presents the flowchart of the proposed method. In the following, the proposed method is implemented and evaluated.

4.4. Diffusion model in the proposed methods

The information diffusion method in social networks is a form that explains how information and contents are distributed in these networks and how different individuals and users influence this information diffusion. These models are usually defined based on human behavior, social network algorithms, and individual's reactions to the information propagated by others. These models help better recognize information diffusion procedures in social networks, and better program and manage information. Information diffusion models in social networks are usually divided into threshold, cascade, trigger, and epidemic models. These models are introduced as follows.

- Threshold models: The information is directly diffused by the individuals, and each individual decides whether the information is diffused among others. This model is based on personal decision-making.
- Cascade models: The individuals are influenced by the diffused information by others, and information is diffused among others.
- Trigger models: The information is diffused automatically by the individual and without individual decision-making. This model is based on automatic processes and algorithms of social networks.

Pidemic models: The information diffusion is like the spread of an epidemic illness, which is quickly diffused in social networks. This model is based on the quick and widespread information in social networks.

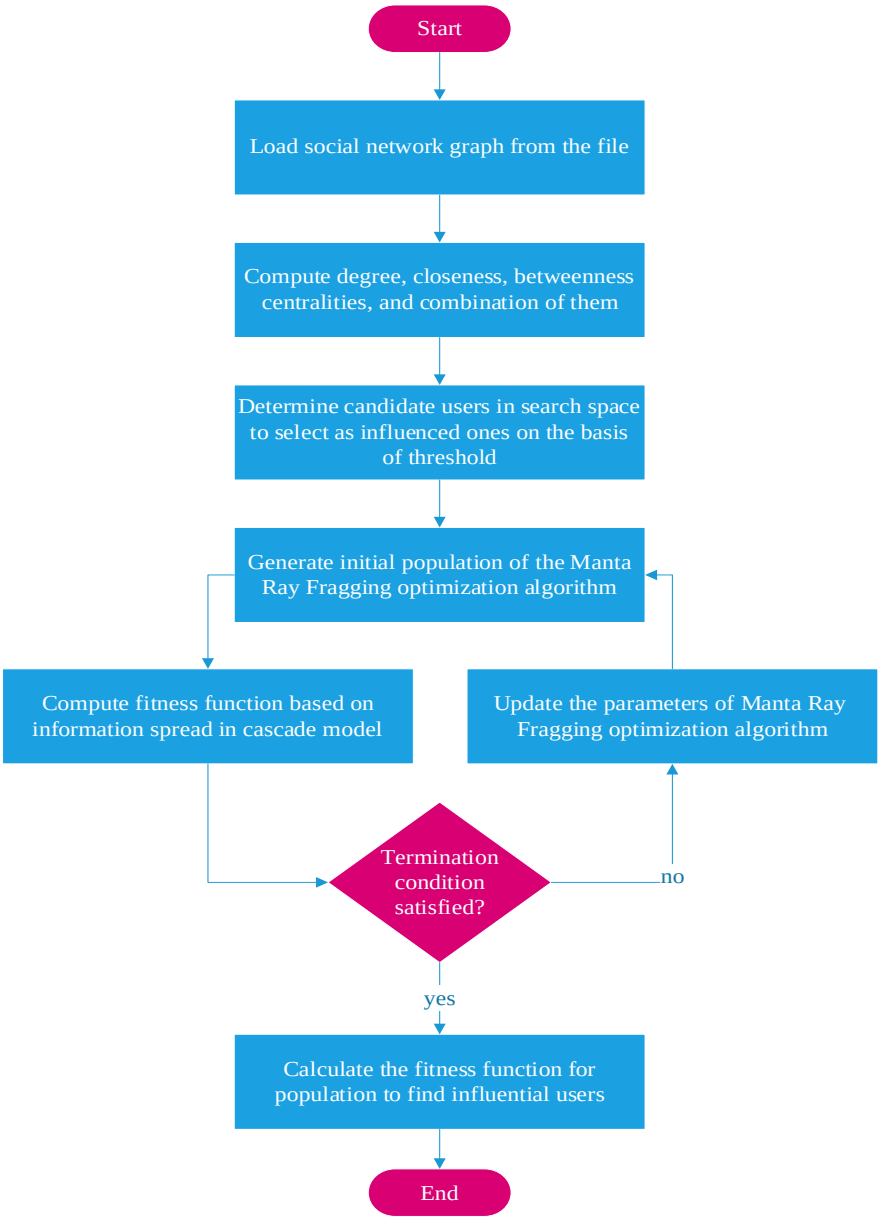


Fig. 7. Flowchart of the proposed method

The cascade diffusion model is used in the proposed model. This model shows the procedure of information transfer and content propagation in social networks. When an individual diffuses content, this content is observed by others, and some of them decide to transfer it to others. This content transfer process continues through individual social communication and is quickly transferred to other individuals on social networks as a cascade. This model shows that individuals are influenced by the contents and information diffusion by others, and this information quickly and widely spreads in social networks. This content diffusion is usually performed due to an individual's psychological and social influences, and it encourages them to transfer and diffuse information influence of others. This model is also useful for checking the effects of content and information propagation in online communities, program marketing strategies, and increasing the influence of social networks. Fig 7. presents the flowchart of the proposed method.

5. IMPLEMENTATION OF THE PROPOSED METHOD

To evaluate the effectiveness and robustness of the proposed method, extensive experiments were conducted using five well-known real-world social network datasets: NetScience, Email, Hamsterster, Ego-Facebook, and Pages-Public Figure, as referenced in [28]. These datasets differ in size, structure, and user interaction patterns, providing a diverse testing ground to verify the generalizability of the proposed approach. Specifically:

- NetScience represents a co-authorship network in scientific publications.
- Email captures email exchanges within a network.
- Hamsterster is a friendship network collected from a pet social website.
- Ego-Facebook contains ego-networks extracted from Facebook profiles.
- Pages-Public Figure includes the connections between verified Facebook pages and public figures.

Each dataset comprises a varying number of nodes (users) and edges (relationships), allowing the method's scalability and adaptability to be tested under different network topologies and densities.

In the simulation process, the proposed method first computes three well-established social importance measures—degree centrality, closeness centrality, and betweenness centrality—for each node within the network. These metrics quantify each user's potential to spread information based on their position and connectivity in the graph.

Subsequently, these centrality measures, individually and in combination, are incorporated into a customized fitness function used by the enhanced Manta-Ray Foraging Optimization (MRFO) algorithm. This fitness function aims to balance two objectives: (1) maximize the overall influence spread and (2) minimize the number of seed users, ensuring an efficient selection of influential nodes.

To initialize the MRFO algorithm, an initial population of candidate seed sets is generated. This population is strategically constrained to reduce computational overhead: candidate users are pre-selected by applying a threshold based on the mean value of a given centrality measure in the network. For example, in the degree centrality scenario, only users whose degree centrality exceeds the network average are considered as potential seeds. This pre-filtering effectively reduces the search space by excluding nodes with minimal influence potential.

During the iterative optimization, the MRFO algorithm explores this reduced solution space. In each iteration, the current population of candidate solutions is evaluated using the bi-objective fitness function. Based on the MRFO's foraging-inspired update rules, a new population is generated by refining the influential user selection to maximize influence spread while maintaining a compact seed set.

This simulation process is repeated for each scenario (degree centrality, closeness centrality, betweenness centrality, and fusion of these measures) across all datasets. The final output is a set of influential users for each network and scenario, along with quantitative results showing how much information spread is achieved relative to other baseline methods.

The parameters in the proposed CMMRFO algorithm have set based on a combination of network structural properties and algorithmic design elements that are inherently sensitive to performance. First, the initial population in the MRFO algorithm was discretized to represent user indices in the social network, with a threshold based on the average degree centrality used to filter users—this ensures that only users with above-average connectivity are considered, improving convergence toward influential nodes. Second, the fitness function incorporates two performance-sensitive parameters: the number of influenced users (spread potential) and the weight of social importance, which is calculated using a weighted combination of degree, closeness, and betweenness centrality. The weights w_1,w_2, and w_3 are bounded between 0 and 1 and directly affect the optimization outcome, making them critical to algorithm sensitivity. Finally, by using the cascade diffusion model, the influence spread is modeled realistically, and the parameterization adapts dynamically to the structure of the network, further enhancing the relevance of parameter choices to actual performance.

In the proposed CMMRFO algorithm, standard MRFO parameters such as the population size, maximum number of iterations, and exploration-exploitation control mechanisms are carefully set to balance search quality and computational cost. The population size determines how many candidate solutions are explored simultaneously, influencing convergence speed and diversity. The maximum iteration limit ensures that the algorithm stops after a reasonable time while allowing enough search depth. Additionally, MRFO's search operators — including Chain Foraging, Cyclone Foraging, and Somersault Foraging — control how candidate solutions update their positions. These operators are adapted to work with discrete user indices, ensuring effective exploration of the social network space. Proper tuning of these parameters helps achieve an optimal trade-off between exploration and exploitation, directly impacting the influence maximization performance. The main

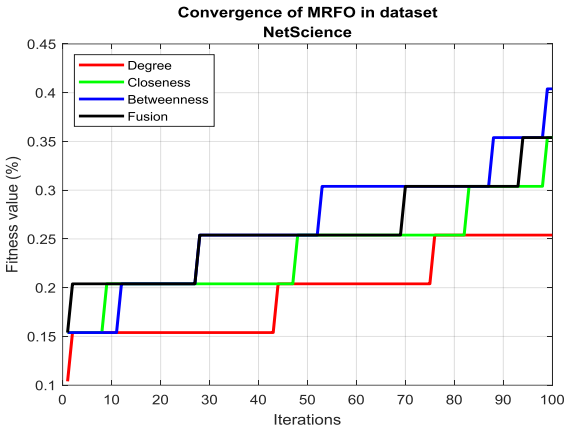
parameters of the proposed method are listed in Table 2.

In addition, the solution having the best fitness function value is selected as the optimal solution. The new population is evaluated based on the fitness function, and the optimal solution is selected. This procedure continues until the stop condition, which is 100 iterations, is met. The last solution related to the last iteration shows that the influenced user's selected generation represents the influencers based on a specified importance criterion. Figs 8 – 12. show a convergence diagram of the optimal point related to each social importance in each data set's MRFO algorithm.

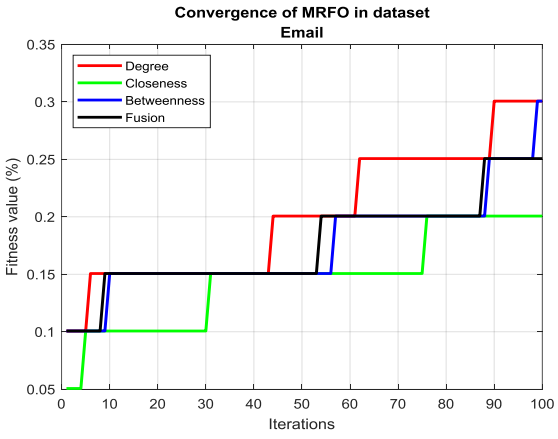
As shown in Figs 8 -12, the convergence diagram of the MRFO algorithm is drawn based on different social importance criteria in the mentioned data sets. As expected, fitness function values are increased in ascending order in each step in the Convergence diagram. Hence, it can be found that the MRFO algorithm does not fall into local traps and converges continuously toward the optimal point. Finally, influenced users are found in a network in the final solution of each scenario. In the following, the proposed method is evaluated.

TABLE 2
Parameters of the proposed method

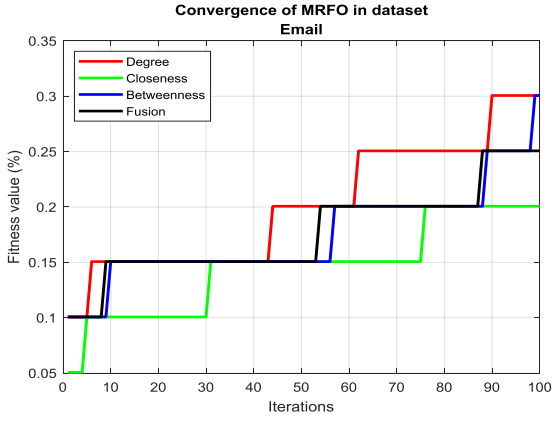
Parameter	**Value**	**Description**
Initial Population	Users with degree centrality above mean	Limits search to influential users, improving convergence and solution quality.
Discretization Method	Continuous-to-integer mapping	Ensures valid user indices; maintains MRFO compatibility with the discrete domain.
Centrality Weights (w_1 , w_2 , w_3)	[0, 1]	Controls the importance of degree, closeness, and betweenness; balances multiple influence aspects.
Selection Threshold	Average degree centrality	Dynamically adjusts candidate pool size; aligns with network density.
Fitness Function	Spread + penalty for the number of seeds.	Encourages high influence spread with minimum seed users; balances cost and benefit.
Diffusion Model	Cascade model	Realistic simulation of influence propagation; validates optimization results.
Population Size (MRFO)	20–50	Balances exploration diversity and computational cost; affects convergence.
Maximum Iterations (MRFO)	100–500	Defines search depth; more iterations can improve solution quality.
MRFO Operators	Chain, Cyclone, Somersault Foraging	Ensure effective exploration and exploitation; adapted for discrete indices.



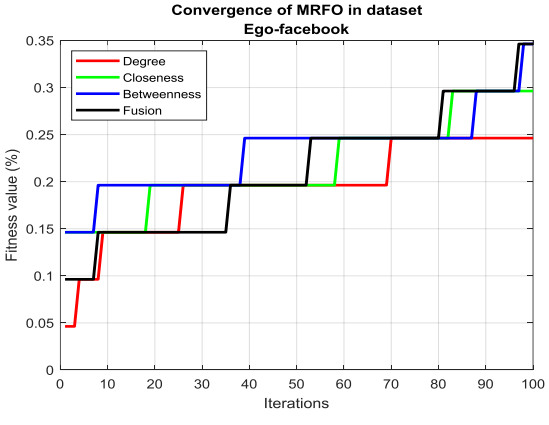
(a)



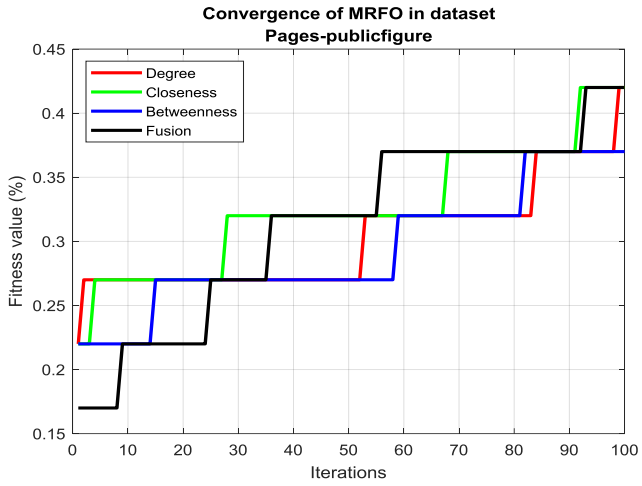
(b)



(c)



(d)



(e)

Fig. 8. Convergence of MRFO based on different social importance criteria in the data set: (a) NetScience, (b) Email, (c) Hamsterster, (d) Ego-facebook, (e) Pages – publicfigure

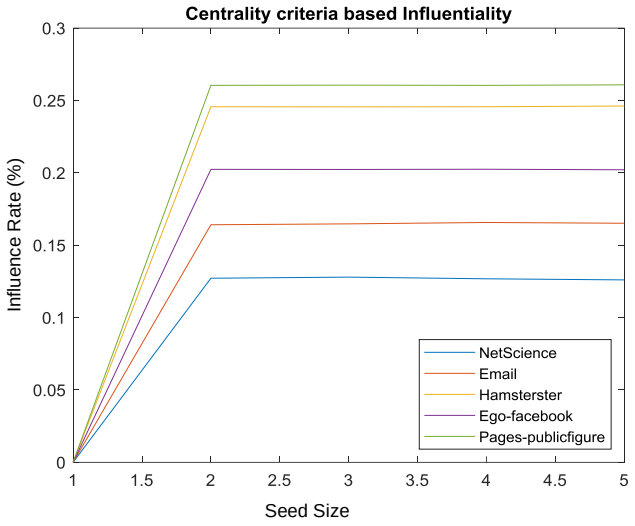


Fig. 9. The influence rate is based on the degree centrality criterion in various data sets

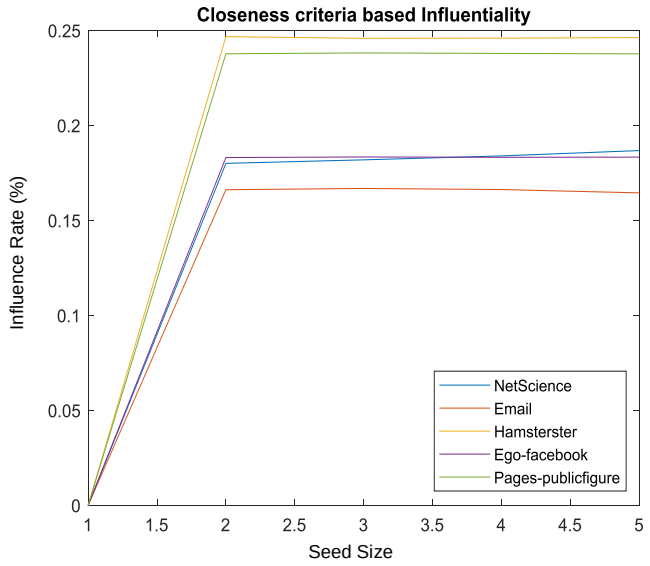


Fig. 10. The influence rate is based on the closeness centrality criterion in different data sets.

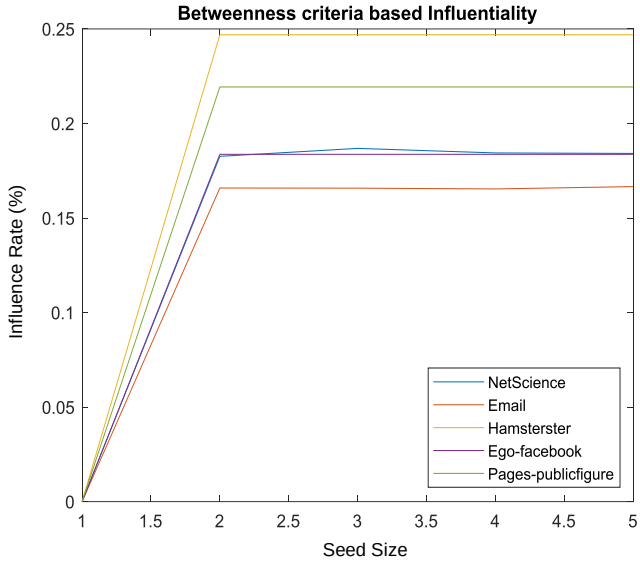


Fig. 11. The influence rate is based on betweenness centrality criteria in different data sets.

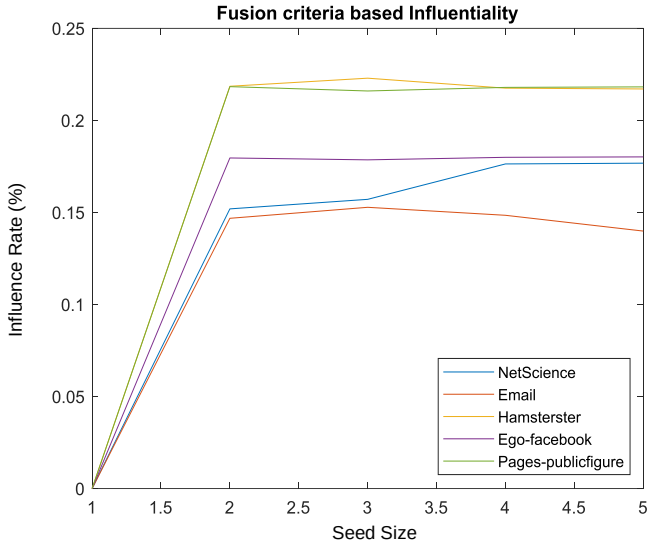


Fig 12. The influence rate is based on a combination of criteria in different data sets

5.1. Evaluation of the proposed method

After implementing the proposed method based on different scenarios, its performance is evaluated. The most conventional criterion is to evaluate the number of users influenced by users in the social network to evaluate the proposed method in terms of IM. In other words, the purpose of IM is to find the users having the most influence among other users.

Therefore, each solution that can find the most influential user can be considered optimal. In the proposed method, the problem of IM is solved by three popular importance criteria, including degree centrality, closeness centrality, betweenness centrality, and a combination of centrality criteria using the CMMRFO algorithm. The proposed method is implemented based on four different scenarios according to importance and combinational importance criteria. In the following, the number of users influenced by influenced users is inspected in each scenario. The number of users influenced by influential users is shown in Table 3. based on social importance criteria in different data sets.

According to Table 1, it can be found that the CMMRFO algorithm influences various users by each social importance criterion for each data set. On the other hand, the proposed IM finds the minimum number of users with the highest influence on the users. The results of IM are more optimal when the effect is high, and the number of influenced users is low. Therefore, the whole number of influenced users is divided by the whole number of users to compute the influence rate for each influenced user. Figs 9-12. show the influence rate of degree centrality, closeness centrality, betweenness centrality, and combination of those centralities in different data sets, respectively.

As is evident in Figs. 9 - 12, different social importance criteria obtain various influence rates. Fig 18 shows a bar graph comparing influence rates based on social importance criteria.

TABLE 3
The number of influenced users in different data sets based on social importance criteria

Datasets	**Centrality**	**Closeness**	**Betweenness**	**Fusion criteria**
NetScience	41	46	36	40
Email	128	120	124	123
Hamsterster	276	280	271	263
Ego-facebook,	620	601	621	617
Pages - publicfigure	1161	1171	1168	1166

Figes 9 to 12 illustrate how the influence rate varies with the number of selected influential users, commonly referred to as the seed size, for each centrality criterion—degree, closeness, betweenness, and their fusion. In these figures, the horizontal axis denotes the seed size, which ranges from 0 to 5, reflecting the predefined and limited number of influential users chosen by the proposed method. The vertical axis shows the influence rate, defined as the ratio of users influenced by these seeds to the total number of users in the network.

A higher influence rate indicates a more effective spread of information initiated by a small number of well-chosen seeds, demonstrating the strength of the selection strategy. As depicted, the influence rate increases rapidly for the first few seeds, highlighting that the initial influential users have the highest impact on spreading information. Subsequently, the growth slows and stabilizes, as additional seeds contribute incrementally less due to network structure saturation and overlap with already influenced nodes.

This trend across Figes 9 to 12. confirms that the proposed CMMRFO algorithm successfully prioritizes users with the most advantageous network positions for rapid and widespread diffusion. The consistency of this pattern for all centrality measures further validates the robustness and adaptability of the seed selection process in various network conditions.

Regarding the trend illustrated in the plots, it can be observed that the influence rate initially increases rapidly during the first few stages and then gradually converges to an approximately stable value. This pattern arises because, in the proportional function of the proposed CMMRFO algorithm, users with the highest centrality parameters are prioritized as influential seeds. This ensures that users with the most significant connections within the network are selected first, resulting in a steep initial rise in the number of directly influenced users. The first two influential users typically contribute to a large portion of the network being directly affected due to their high connectivity and central position within the network structure.

As the algorithm proceeds to select additional influential users, it ensures minimal overlap with the initially chosen seeds to avoid redundant influence spread. Consequently, each additional user contributes fewer new connections than the initial seeds, leading to a decrease in the growth rate of the influence rate. This behavior explains why, after selecting the first two influential users, the increase in the influence rate diminishes and gradually levels off to a near-constant value. This convergence demonstrates the efficiency and precision of the proposed CMMRFO-based selection strategy in maximizing the spread of influence with an optimal and minimal number of influential users.

Fig 13. illustrates the comparative performance of the proposed CMMRFO algorithm when applying different centrality criteria for selecting influential users. Specifically, this figure highlights how the choice of centrality measure—degree, closeness, betweenness, or their fusion—affects the spread of information within the network. The plotted influence rates demonstrate that selecting seeds based on degree centrality consistently results in a higher influence spread than using closeness, betweenness, or their combination.

This result can be explained by the inherent advantage of degree centrality: nodes with higher degrees have more direct links, allowing information to propagate rapidly to a larger portion of the network in the initial diffusion stages. In contrast, closeness and betweenness centralities, while valuable for understanding network structure, may select nodes that are strategically positioned but have fewer immediate connections, leading to a slower initial spread.

Therefore, Fig 13. validates the design decision in the CMMRFO framework to prioritize degree centrality within its hybrid selection mechanism. By doing so, the algorithm effectively balances structural awareness and computational efficiency, ensuring that the selected influential users trigger faster and broader diffusion compared to other criteria. This empirical comparison further supports the claim that the integration of social network metrics into the MRFO algorithm provides a robust solution for the influence maximization problem in large-scale networks.

As shown, Degree Centrality consistently achieves a higher influence rate than the other measures in various network datasets. This highlights that the CMMRFO algorithm, when guided by the degree centrality parameter, is more effective in identifying highly connected nodes that maximize the spread of information throughout the network.

Including this comparison was necessary to validate why the Degree Centrality parameter was given more weight in our hybrid selection mechanism within the CMMRFO framework. It supports our motivation for favoring nodes with higher direct connections to achieve faster and broader information dissemination.

The critical point is that the maximum number of users influenced by influenced users is selected based on social importance criteria and the CMMRFO algorithm. The purpose is to increase the number of users connected with these users directly and use information diffusion directly as influenced by neighbor users. The influence is maximized when many influenced users select the minimum number of users with maximum influence on other selected users, which is introduced as the optimal

method. Therefore, it is necessary to compare the number of users influenced by influenced users in the proposed method and [6] by implementing the moth flame optimization algorithm without using social importance criteria and a fixed number of influenced users. The maximum number of users influenced by the CMMRFO algorithm is shown in Fig 19. and compared with the primary method.

According to Fig 14. it is evident that incorporating social importance criteria—particularly degree centrality—into the MRFO algorithm significantly improves the overall influence spread in social networks. This figure presents a comparative bar chart showing the influence rates achieved by different centrality-based strategies. Among them, the approach that relies on degree centrality consistently outperforms closeness, betweenness, and fusion criteria across multiple datasets. This improvement is attributed to the fact that users with a high degree centrality typically maintain a large number of direct connections, enabling rapid and broad diffusion of information.

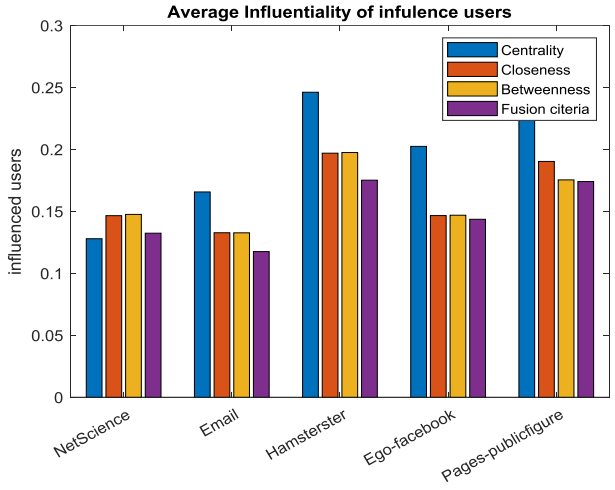


Fig. 13. Bar graph comparing influence rate based on social importance criteria.

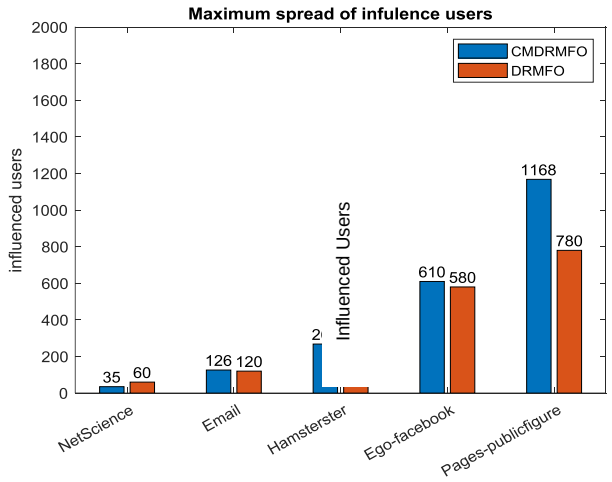


Fig. 14. The comparison of the maximum number of influence users in different data sets

Furthermore, the proposed method not only enhances the spread of influence but does so using a fixed and minimal number of seed users, highlighting its efficiency. By combining optimized fitness function parameters with centrality-based node selection, the CMMRFO algorithm ensures that influence is maximized without redundancy or overlap in the selection of seed users. This results in a higher influence rate, defined as the number of influenced users relative to the total number of seeds, compared to traditional methods. These results confirm that integrating structural characteristics of the network into the optimization process yields a more effective and scalable solution to the influence maximization problem. Fig15. compares the influence rate in the presented method and others.

According to Fig 15. it is evident that the proposed CMMRFO algorithm achieves a significantly higher influence rate compared to the existing baseline method that does not incorporate social importance criteria. This demonstrates the effectiveness of integrating the degree centrality criterion into the fitness function, as it prioritizes highly connected nodes that can spread information more rapidly and extensively. The figure illustrates that for each dataset, the influence rate increases noticeably when the centrality-based approach is used, confirming that selecting seed users based on structural importance leads to a more efficient diffusion process. Moreover, the results highlight that the improvement is most pronounced in networks with highly heterogeneous structures, where influential nodes play a critical role in connecting distant parts of the network. Overall, these findings validate that employing social importance measures, particularly degree centrality, substantially enhances the performance of influence maximization algorithms by ensuring a higher spread of information with fewer seed users.

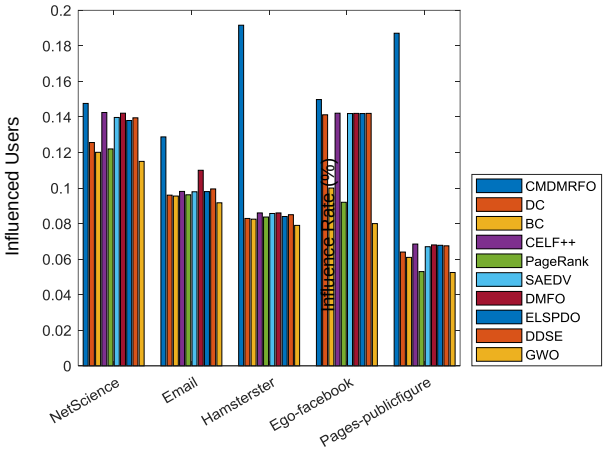


Fig. 15. The comparison of the influence rate in the proposed method and other methods

6. Conclusion and Future Work

IM is the process of selecting a small set of nodes to ensure the quickest and widest information diffusion in social networks. Detecting such nodes remains a crucial research topic with numerous practical applications. While greedy-based methods provide reliable solutions, their high computational cost due to extensive Monte Carlo simulations makes them unsuitable for large-scale networks. In contrast, structural centrality-based approaches offer an efficient alternative by leveraging the inherent properties of network graphs.

In this paper, a novel discrete version of the Manta Ray Foraging Optimization (MRFO) algorithm was developed specifically for the IM problem. The proposed method integrates three prominent centrality measures—degree, closeness, and betweenness—into a fused fitness function and applies innovative discretization techniques to handle the discrete search space effectively.

Experimental evaluations on five real-world social network datasets demonstrated that the proposed CMMRFO algorithm consistently achieves superior influence spread compared to conventional methods. Notably, it yielded average improvements of 14.63% for NetScience, 12.81% for Email, 19.03% for Hamsterster, 15.24% for Ego-Facebook, and 18.76% for Pages-PublicFigure networks. These significant improvements highlight the robustness, scalability, and practical effectiveness of the proposed approach in maximizing influence with a minimal seed set.

The key development lies in designing a bi-objective fitness function that balances two essential goals: minimizing the number of influential (seed) users and simultaneously maximizing their impact. This balance ensures not only a high influence spread but also cost-effective targeting strategies. Furthermore, the integration of fused centrality measures empowers the algorithm to exploit complementary structural information, resulting in a more accurate identification of influential nodes.

In summary, the main contributions of this work can be highlighted as follows:

- A novel discrete MRFO algorithm adapted for the discrete IM problem.
- Introduction of a fused centrality index combining degree, closeness, and betweenness to guide seed selection more effectively.
- Development of a bi-objective fitness function that optimally balances the number of seeds and influences spread.
- Comprehensive evaluation demonstrating significant influence spread improvements over baseline algorithms across various real-world networks.

These contributions together advance the state-of-the-art in influence maximization, especially for large and complex social networks.

For future research, the proposed method can be extended by integrating a community detection step alongside the centrality-based user selection. This hybrid approach would allow meta-heuristic algorithms to identify key influencers within each detected community, combining global and local structural insights. Such an extension is expected to further enhance the efficiency, precision, and adaptability of the model for large-scale and highly modular social networks.

7. REFERENCES

[1] Z. Aghaee, M. M. Ghasemi, H. A. Beni, A. Bouyer, and A. Fatemi. (2021, Mar.). A Survey on Meta-Heuristic Algorithms for the IM Problem in Social Networks. *Computing*. [Online]. pp. 2437–2477. Available: https://doi.org/10.1007/s00607-021-00945-7

[2] Y. Ye, Y. Chen, and W. Han. (2022, Oct.). IM in social networks: theories, methods and challenges. *Array*. [online]. *16*, p. 100264. Available: https://doi.org/10.1016/j.array.2022.100264

[3] Z. Aghaee and S. Kianian. (2020, Apr.). IM algorithm based on reducing search space in the social networks. *SN Applied Sciences* .[online]. *2*, pp. 1-14. Available: https://doi.org/10.1007/s42452-020-03812-w

[4] P. Dey, A. Chatterjee, and S .Roy. (2019, Jul.). IM in online social network using different centrality measures as seed node of information propagation. *Sādhanā Department of Computer Science and Engineering*. [online]. *44(9)*, pp. 1-13. Available: https://doi.org/10.1007/s12046-019-1189-7

[5] L. Wang, L. Ma, C. Wang, N.-g. Xie, J. M. Koh, and K. H. Cheong. (2021, May.). Identifying influential spreaders in social networks through discrete moth-flame optimization. *IEEE Transactions on Evolutionary Computation*. [online]. *25(6)*, pp. 1091-1102. Available: http://dx.doi.org/10.1109/TEVC.2021.3081478

[6] L. C. Freeman. (1978, Sep.). Centrality in social networks: Conceptual clarification. *Social Networks*. [Online]. *1(3)*, pp. 215–239. Available: https://dx.doi.org/10.1016/0378-8733(78)90021-7

[7] D. Kempe, J. Kleinberg, and É. Tardos. (2003, Aug.). Maximizing the spread of influence through a social network. Presented at Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 137–146. [Online]. Available: https://doi.org/10.1145/956750.956769

[8] S. Brin and L. Page. (1998, Apr.). The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*. [Online]. *30(1–7)*, pp. 107–117. Available: https://dx.doi.org/10.1016/S0169-7552(98)00110-X

[9] Q. Jiang, G. Song, C. Gao, Y. Wang, W. Si, and K. Xie. (2011, Aug.). Simulated annealing based influence maximization in social networks. Presented at Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence. [Online]. 25(1), pp. 127–132. Available: https://doi.org/10.1609/aaai.v25i1.7838

[10] A. Şimşek and R. Kara. (2018, Dec.). Using swarm intelligence algorithms to detect influential individuals for influence maximization in social networks. *Expert Systems with Applications*. [Online]. *114*, pp. 224–236. Available: https://doi.org/10.1016/j.eswa.2018.07.038

[11] Y. D. Navaei, M. H. Rezvani, and A. M. E. Moghaddam. (2024, Feb.). A novel neighborhood-based importance measure for social network influence maximization using NSGA-III. Presented at 2024 10th International Conference on Artificial Intelligence and Robotics (QICAR), pp. 113-118. [Online]. Available: https://doi.org/10.1109/QICAR61538.2024.10496642

[12] L. Cui, H. Hu, S. Yu, Q. Yan, Z. Ming, Z. Wen, and N. Lu. (2018, Feb.). DDSE: A novel evolutionary algorithm based on degree-descending search strategy for influence maximization in social networks. *Journal of Network and Computer Applications*. [Online]. *103*, pp. 119–130. Available: https://doi.org/10.1016/j.jnca.2017.12.003

[13] J. Tang, R. Zhang, Y. Yao, F. Yang, Z. Zhao, R. Hu, and Y. Yuan. (2019, Jan.). Identification of top-k influential nodes based on enhanced discrete particle

swarm optimization for influence maximization. *Physica A: Statistical Mechanics and Its Applications*. [Online]. 513, pp. 477–496. Available: https://doi.org/10.1016/j.physa.2018.09.040

[14] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. (2007, Aug.). Cost-effective outbreak detection in networks. in Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 420-429. [online]. Available: https://doi.org/10.1145/1281192.1281239

[15] S. Mokhtarizadeh, B. Zamani Dehkordi, M. Mosleh, and A. Barati. (2021, Oct.). Influence maximization using time delay based harmonic centrality in social networks. *Tabriz Journal of Electrical Engineering*. [Online]. pp. 359–370.

[16] L. Han, Q. Zhou, J. Tang, X. Yang, and H. Huang. (2021, Feb.). Identifying top-k influential nodes based on discrete particle swarm optimization with local neighborhood degree centrality. *IEEE Access*. [Online]. *9*, pp. 21345–21356. Available: https://doi.org/10.1109/ACCESS.2021.3056087

[17] A. Arora, S. Bansal, C. Kandpal, R. Aswani, and Y. Dwivedi. (2019, Jul.). Measuring social media influencer index—insights from Facebook, Twitter and Instagram. *Journal of Retailing and Consumer Services*. [Online]. *49*, pp. 86–101. Available: http://dx.doi.org/10.1016/j.jretconser.2019.03.012

[18] B. Fu, J. Zhang, W. Li, M. Zhang, Y. He, and Q. Mao. (2022, Jul.). A differential evolutionary influence maximization algorithm based on network discreteness. *Symmetry*. [Online]. *14(7)*, p. 1397. Available: https://doi.org/10.3390/sym14071397

[19] B. Chatterjee, T. Bhattacharyya, K. K. Ghosh, A. Chatterjee, and R. Sarkar. (2021, Feb.). A novel meta-heuristic approach for influence maximization in social networks. *Expert Systems*. [Online]. p. 12676. Available: https://doi.org/10.1111/exsy.12676

[20] Tarun K. Biswas , A. Abbasi, Ripon, K. Chakrabortty , "An MCDM integrated adaptive simulated annealing approach for influence maximization in social networks", vol.556, pp. 27-48, (2021). Availabe: https://doi.org/10.1016/j.ins.2020.12.048

[21] H. Li, R. Zhang, Z. Zhao, X. Liu, and Y. Yuan. (2021, Nov.). Identification of top-k influential nodes based on discrete crow search algorithm optimization for influence maximization. *Applied Intelligence*. [Online]. *51(11)*, pp. 7749–7765. Available: http://doi.org/10.1007/s10489-021-02283-9

[22] R. Cantini, F. Marozzo, S. Mazza, D. Talia, and P. Trunfio. (2021, Sep.). A weighted artificial bee colony algorithm for influence maximization. *Online Social Networks and Media*. [Online]. *26*, p. 100167. Available: https://doi.org/10.1016/j.osnem.2021.100167

[23] A. Karczmarczyk, J. Jankowski, and J. Wątrobski. (2021, Jun.). Multi-criteria seed selection for targeted influence maximization within social networks. Presented at International Conference on Computational Science (ICCS 2021), pp. 454–461. [Online]. Available: https://doi.org/10.1007/978-3-030-77967-2-38

[24] L. Han, K. C. Li, A. Castiglione, J. Tang, H. Huang, and Q. Zhou. (2021, Apr.). A clique-based discrete bat algorithm for influence maximization in identifying top-k influential nodes of social networks. *Soft Computing—A Fusion of Foundations, Methodologies & Applications*. [Online]. *25(13)*, pp. 8223–8240. Available: https://doi.org/10.1007/s00500-021-05749-7

[25] W. Zhao, Z. Zhang, and L. Wang. (2020, Jan.). Manta ray foraging optimization: An effective bio-inspired optimizer for engineering applications. *Engineering Applications of Artificial Intelligence*. [Online]. *87*, p. 103300. Available: https://doi.org/10.1016/j.engappai.2019.103300

[26] S. Wang, Y. Wang, Y. Wang, and Z. Wang. (2022, Oct.). Comparison of multi-objective evolutionary algorithms applied to watershed management problem. *Journal of Environmental Management*. [Online]. *324*, p. 116255. Available: https://doi.org/10.1016/j.jenvman.2022.116255

[27] R. R. Singh. (2021, Aug.). Centrality measures: A tool to identify key actors in social networks. In *Principles of Social Networking: The New Horizon and Emerging Challenges*. [Online]. pp. 1–27. Available: https://doi.org/10.1007/978-981-16-3398-0-1

[28] R. Rossi and N. Ahmed. (2015, Jan.). The network data repository with interactive graph analytics and visualization. in Proceedings of the AAAI conference on artificial intelligence. [online]. Available: https://doi.org/10.1609/aaai.v29i1.9277

[29] S. Ghafori and F. S. Gharehchopogh. (2022, Jan.). A multiobjective Cuckoo Search Algorithm for community detection in social networks. In *Multi-objective Combinatorial Optimization Problems and Solution Methods*. [Online]. pp. 177–193. Available: https://doi.org/10.1016/B978-0-12-823799-1.00007-3

[30] A. Heydariyan, F. S. Gharehchopogh, and M. R. E. Dishabi. (2024, Jul.). A hybrid multi-objective algorithm based on slime mould algorithm and sine cosine algorithm for overlapping community detection in social networks. *Cluster Computing*. [Online]. *27(10)*, pp. 13897–13917. Available: https://doi.org/10.1007/s10586-024-04632-y

[31] J. Sheykhzadeh, B. Zarei, and F. S. Gharehchopogh. (2024, Jul.). Community detection in social networks using a local approach based on node ranking. *IEEE Access*. [Online]. *12*, pp. 92892–92905. Available: https://doi:10.1109/ACCESS.2024.3420109

[32] S. M. Aghdam, F. S. Gharehchopogh, and M. Masdari. (2024, Mar.). A hybrid approach in opinion leaders selection using African vultures optimization and hunger games search algorithms. *Social Network Analysis and Mining*. [Online]. *14(1)*, p. 60. Available: https://doi.org/10.1007/s13278-024-01228-7

[33] S. M. Aghdam, F. S. Gharehchopogh, and M. Masdari. (2025, Apr.). An opinion leader selection in social networks using hybrid amended salp swarm and improved grey wolf optimizer algorithms. *International Journal of Data Science and Analytics*. [Online]. pp. 1–26. Available: https://doi.org/10.1007/s41060-025-00760-9

Journal of Computer & Knowledge Engineering Vol. 8, No.2, 2025 Serial No.: 16

CONTENTS