



Fuzzy Adaptive Bobcat Optimization Algorithm (FABOA) Feature Selection Model and Double Deep Q Network (DDQN) Learning for Big Data Analytics

M. Vijetha ^{a,*}, G. Maria Priscilla ^a

^a Department of Computer Science, Sri Ramakrishna College of Arts and Science, Coimbatore, Tamil Nadu, India.

* Corresponding Author Email: vijethachandran1983@gmail.com

DOI: <https://doi.org/10.54392/irjmt2661>

Received: 19-01-2026; Revised: 18-09-2026; Accepted: 12-10-2026; Published: 21-10-2026



Abstract: Big data analytics is challenged by high dimensionality, redundant attributes, heterogeneous data distributions, and the computational burden of processing large-scale datasets. The current filter, wrapper and matrix factorization approaches typically ignore the relationships among features or converge too early, are not very adaptive, and are not very efficient in searching. To fill this space, this study offers an Intelligent Feature Selection and Large-Scale Data Analytics Framework based on Fuzzy Adaptive Bobcat Optimization Algorithm–Double Deep Q-Network (FABOA–DDQN). In addition to the Bobcat Optimization Algorithm, FABOA introduces triangular fuzzy membership functions to adjust exploration and exploitation based on the diversity of the population, improvement in fitness and the number of iterations. Its wrapper objective takes into account predictive performance, feature-subset compactness and feature redundancy. DDQN is not just a sequential combination, but rather a policy-learning controller: Candidate feature subsets are states, adding or removing features or changing their values is an action, and the reward value is a measure of fitness improvement and feature reduction. The online and target networks alleviate Q-value overestimation and stabilize search decisions. The experimental results were obtained by applying a 70:15:15 training/validation/testing split of a widely used SVM evaluator, 30 independent trials and tested against NMF, S4NMF, RLNMFA, DBMD variants, BOA, GA, PSO, WOA, DQN, and DDQN. On Covertype, FABOA–DDQN achieved 98.34% precision, 96.77% recall, 97.55% F1-score, and 97.10% RI. It surpassed DBMD–DLM–BFGS in all four of these measures by as much as 1.95%, 3.44%, 2.69%, and 3.29%, respectively, across all four datasets while bringing the ratio of the selected features down to 31%.

Keywords: Big Data Analytics, Feature Selection, Fuzzy Adaptive Bobcat Optimization Algorithm (Faboa), Double Deep Q Network (Ddqn), Clustering Optimization, Evaluation Metrics.

1. Introduction

Large, diverse, and high-dimensional datasets are being generated by the fast growth of IoT, cloud computing, social media, and intelligent sensing technologies. Data-intensive applications including healthcare, cybersecurity, financial analytics, and smart-city management use such data. However, irrelevant, redundant, and noisy features might raise processing needs, search space, and analytical model generalisation. Feature selection (FS) is a key preprocessing approach for discovering valuable properties and removing superfluous features. To handle huge and complicated feature spaces, optimization-driven FS has been studied further. Through learnt choice mechanisms, reinforcement-learning-guided optimisation may autonomously navigate feature-subset spaces [1]. Reinforcement learning using particle swarm optimisation and adaptive

scoring has enhanced high-dimensional FS search [2], while evolutionary Q-learning has improved optimization-based feature selection [3]. Online feature-subset selection utilising incremental learning and evolutionary computing accommodates developing feature properties for continually created data [4]. Bowerbird-inspired search methods [5] and improved grey wolf optimisation algorithms [6] have been used to high-dimensional FS. These techniques enhance complicated feature space exploration, although population-based optimisation may still demand a lot of computer resources to analyse many potential feature subsets. The process used to establish feature relevance classifies feature-selection strategies as filter, wrapper, and embedding. Filter approaches use statistical or information-theoretic criteria and are computationally efficient, but their assessment is independent of the downstream learning model and may not capture nonlinear feature interactions. Wrapper

approaches analyse potential subsets based on a learning algorithm's performance to find feature combinations that match the learning target. However, for high-dimensional datasets, wrapper FS might be computationally costly due to downstream learner training and assessment. Wrapper optimisation efficiency and resilience have been the subject of recent study. Hierarchical learning and modified teaching-learning-based optimisation have been studied to improve wrapper FS exploration and exploitation [7]. An adaptive and parallel Arctic Puffin Optimisation metaheuristic system accelerates wrapper-based feature selection and addresses high-dimensional search computing needs [8]. These results show that wrapper FS needs high-quality subset identification, computational complexity control, and search efficiency. Advanced metaheuristic and reinforcement-learning methods have been added to feature-selection optimisation. In high-precision wrapper FS, quantum-inspired jellyfish search optimisation improves discriminative feature subset identification [9]. To improve feature selection adaptive search, growth-starfish optimisation has been combined with opposition-based and reinforcement learning [10]. To improve feature-selection transparency while maintaining optimisation, explainable statistical reinforcement learning has been investigated [11]. Deep reinforcement learning and Grey Wolf Optimisation for hyperspectral picture band selection show that learned decision rules can solve high-dimensional selection issues [12]. These results show that reinforcement learning may dynamically adjust optimisation choices to the search state to augment standard metaheuristic operators. Existing techniques mostly tweak metaheuristics, provide specialised search operators, or use reinforcement learning in particular application areas. A framework that blends adaptive metaheuristic exploration control with learnt value-based feature-selection judgements is available for study. Application-specific high-dimensional learning issues emphasise robust FS. Metaheuristic-driven FS combines dimensionality reduction with classification effectiveness to enhance cancer classification [13]. To increase complicated feature space exploration in wrapper-based FS in high-dimensional medical datasets, enhanced Harris Hawks Optimisation has been studied [14]. In dynamic learning contexts, meta-learning-based multi-view and multi-label FS accommodates streaming labels and changing feature-label connections [15]. To reduce search redundancy and preserve meaningful characteristics, differential-evolution-based FS uses individual reduction and symmetric-uncertainty-based attribute recovery [16].

This paper presents a hybrid intelligent framework employing FABOA for wrapper-based feature selection and DDQN for adaptive learning to overcome existing limits. The BOA uses bobcats' predatory and adaptable hunting to explore globally and exploit locally.

BoA search parameters are continually adjusted using a fuzzy adaptive technique to improve convergence stability and prevent standstill. DDQN uses enhanced BOA attributes for more accurate and robust classification and clustering. The FABOA-DDQN system outperforms distributed optimization and deep learning in precision, recall, F1-score, and RI on high-dimensional UCI Machine Learning Repository datasets. For next-generation big data analytics, this hybrid technique is scalable, intelligent, and efficient.

Instead of a straightforward sequential usage of an adaptive metaheuristic search dynamics and reinforcement-learning based policy control, the novelty of FABOA-DDQN is the integration of both algorithms. FABOA generates various candidate regions and dynamically schedules exploration and exploitation, and DDQN learns which feature transformations are likely to yield long-term improvements. This design is intended to address the drawbacks of fixed-parameter metaheuristics, as well as approach relying on computationally intensive wrapper search and the approach based on DQN which is not specifically developed for compact feature-subset optimization. Evaluated on the Covtype, KDD-99 and SUSY datasets with the same experimental protocol as well as compared with representative matrix-factorization, distributed optimization, metaheuristic and RL-based feature-selection methods. The assessment is based on the improvement of predictive performance, feature reduction and convergence stability of the integration of heterogeneous large-scale datasets. The main contribution is an adaptive, policy-driven feature-selection framework to enhance accuracy, compactness and computational efficiency in big data analysis.

2. Literature Review

NMF-based missing value (NA) imputation in MS-based metabolomics data was suggested by Xu *et al.* (2021) [17]. Rebuilding incomplete datasets using global and local data. Compare it against k-Nearest Neighbours (kNN), Random Forest (RF), and Outlier-Robust Imputation (ORI) on imputation accuracy, data structure retrieval, and rank superiority. The NMF-based technique outperformed kNN and ORI in outlier robustness and imputation accuracy on MS-based datasets, performing similarly to RF. Das *et al.* (2023) [18] suggested this DBMD model for large data clustering and mining. Distribution efficiency was improved by using Statistical Inference, ADMM, and AGD optimization frameworks. Synthetic and real-world data trials verified theoretical convergence analysis. DBMD surpassed S-NMF and Scalable k-means++ in scalability and clustering performance, according to Precision, Recall, F-measure, and Rand Index (RI).

A Hierarchical Hybrid Network was used by Zhou *et al.* (2020) [19] to implement Deep Correlation Mining in diverse large data settings. The model analysed multi-type entity associations on DBLP and ResearchGate datasets. We quantified network dependencies using correlation measurements, including intra-layer and inter-layer impact measures. The Intelligent Router uses Deep Reinforcement Learning (DRL) and an upgraded Random Walk with Restart (RWR) algorithm to optimize routing choices. Experimental results showed that the HHN-based approach supported user participation and recommendation accuracy in academic big data systems. Deep Q-Network (DQN)-based feature selection for multisourced IoT data cleaning was suggested by Wang *et al.* (2020) [20].

A DQL model for dynamic network intrusion detection was developed by Alavizadeh *et al.* [21]. The model was trained on NSL-KDD and optimized for discount factor and episode count. Performance was measured by accuracy, precision, recall, and F1-score. In 250 training sessions, 0.001 discount factor yielded the best identification rates. Machine learning lost intrusion classification to DQL. Du *et al.* (2020) [22] presented MonkeyKing, a large data platform adaptive parameter optimization system. The framework used Deep Reinforcement Learning (DRL) with the Deep Q-Network (DQN) structure and its improved versions: Double, Dueling, and Double-Dueling. Apache Spark workload settings were dynamically modified. Optimized Spark dataset performance showed a 25% increase in work efficiency. Execution time, throughput, and resource consumption verified adaptive optimization.

Md Mujeeb *et al.* suggested a deep learning-integrated MapReduce Framework (MRF) for large data categorization (2021) [23]. Adaptive E-Bat Algorithm, which combines EWMA and BA, was used for feature selection. Deep Belief Networks (DBNs) trained using Adaptive E-Bat optimization classified. In unbalanced big data experiments, it achieved 0.8998 accuracy and 0.9144 TPR, proving its efficacy in large-scale classification. For Directed Acyclic Graph (DAG) job scheduling, Chen and Wu (2024) [24] developed an MDP-based scheduling model optimized using an Improved Deep Q-Network (DQN). To improve model stability and convergence, a gradient strategy approach was used. Recall, Training Time, and Execution Efficiency were assessed on synthetic and actual scheduling datasets [25]. The hybrid method outperformed DQN (0.79) and gradient-based scheduling (0.83) with a recall rate of 0.96 after 500 jobs [26]. The findings showed the model's large-scale task optimization efficiency [27-29].

Subhash Kamble *et al.* [30] suggested the Wrapper-Based Feature Selection for Big Data Using a Multi-Core Environment. Since processing huge data is hard, there have been many efforts to design effective

models for feature selection in big data applications. This chapter proposes thorough wrapper-based feature selection. Parallelizing the wrapper feature selection technique on a multi-core CPU. With the KNN classifier, Euclidean separation matrices provide optimum answers. A more cost-effective and dependable large data feature selection approach is provided. The suggested work improves feature selection to match classification accuracy and computation speed of the old methodology. Effective contribution goals include assessing six dataset outcomes' correctness and increasing accuracy with fewer features.

María José Basgall *et al.* [31] proposed the Fast Data Reduction Recommendation Tool for Tabular Big Data Classification Problems (FDR²-BD). The concept uses vertical and horizontal data analysis to combine feature selection to build dense data clusters with uniform sampling reduction to maintain just a few representative samples from each issue area. Its key benefit is keeping the model's predicted quality within a user's threshold. A k-fold hyper-parametrization method that examines all data makes it resilient. Upgraded Apache Spark parallel operations boost speed and scalability. There is substantial experimental study on 25 huge datasets with various features. Reduction rates are usually over 95%, beating state-of-the-art systems like FCNN_MR, which rarely reach 70%. For best results, maintain data representativeness with quality prediction values around 1% of the baseline. To summarise current techniques, see Table 1. The S4NMF framework for data clustering by Chavoshinejad *et al.* [32] integrates self-supervised and semi-supervised learning into the nonnegative matrix factorisation paradigm. Using labelled and unlabelled samples, the technique learns more discriminative latent feature representations while retaining data geometry. Supervised clustering improves accuracy and robustness, especially for datasets with limited labelled information, but its optimisation process relies on fixed matrix factorisation updates and does not address adaptive feature subset selection in high-dimensional big data environments. Robust Local-Coordinate Nonnegative Matrix Factorisation with Adaptive Graph (RLNMFAG) by Tang and Feng [33] improves clustering performance by designing an adaptive graph to capture local neighbourhood relationships and using robust local-coordinate learning to reduce noise and outliers. The adaptive graph regularisation preserves the manifold structure better than NMF-based approaches and improves clustering. Nevertheless, RLNMFAG primarily focuses on latent representation learning through deterministic optimization and does not incorporate reinforcement learning or adaptive metaheuristic search for selecting the most informative feature subsets.

Table 1. Summary of the Existing Methods

Ref	Author (Year)	Methodology	Type	Optimization Strategy	Dataset/Application	Metrics	Key Contribution	Limitations
[17]	Xu <i>et al.</i> (2021)	NMF-based imputation	Matrix Factorization	Global + Local Reconstruction	MS metabolomics	Accuracy, Rank	Robust missing value handling	Limited to imputation; not suitable for feature selection or high-dimensional decision optimization
[18]	Das <i>et al.</i> (2023)	DBMD	Clustering	ADMM, AGD	Big data clustering	Precision, Recall, F1, RI	Scalable distributed clustering	Focused on clustering; lacks feature selection capability
[19]	Zhou <i>et al.</i> (2020)	HHN + DRL	Deep Hybrid	RL-based routing	DBLP, ResearchGate	Accuracy	Correlation mining framework	Not designed for feature subset optimization
[20]	Wang <i>et al.</i> (2020)	DQN-based FS	Reinforcement Learning	Space Search + DQN	IoT data cleaning	Accuracy, Time	RL-based feature selection	High computational cost; limited scalability on very large datasets
[21]	Alavizadeh <i>et al.</i> (2022)	DQL model	Reinforcement Learning	Policy optimization	NSL-KDD	Accuracy, Precision, Recall	RL-based intrusion detection	Task-specific; not generalized for feature selection
[22]	Du <i>et al.</i> (2020)	Monkey King (DQN variants)	RL Optimization	Double/Dueling DQN	Spark workloads	Time, Throughput	Adaptive parameter tuning	Focus on system tuning rather than feature selection
[23]	Md Mujeeb <i>et al.</i> (2021)	E-Bat + DBN	Hybrid Metaheuristic	Bat + EWMA	Big data classification	Accuracy, TPR	Hybrid FS + DL model	Limited adaptability; no learning-based policy guidance
[24]	Chen & Wu (2024)	Improved DQN	RL Optimization	Gradient + DQN	DAG scheduling	Recall, Efficiency	Efficient task scheduling	Not applicable to feature selection problems
[30]	Kamble <i>et al.</i> (2023)	Wrapper FS	Wrapper Method	Multi-core parallelism	Big data FS	Accuracy	Parallel wrapper-based FS	High computational overhead; lacks adaptive search control
[31]	Basgall <i>et al.</i> (2021)	FDR2-BD	Data Reduction	Sampling + FS	25 datasets	Reduction %, Accuracy	High reduction efficiency	May lose fine-grained feature interactions

A threshold-adaptation mechanism has been investigated for wrapper-based evolutionary feature selection, where adaptive threshold control is used to regulate the conversion of continuous search information into feature-selection decisions and improve the efficiency of evolutionary exploration [34]. Such adaptive decision mechanisms are particularly relevant to high-dimensional wrapper optimization because

inappropriate or fixed selection thresholds can lead to either excessive feature retention or premature elimination of potentially informative attributes. A hybrid filter-wrapper strategy has also been proposed for high-dimensional omics data, combining preliminary filtering with wrapper-based optimization to reduce the search space while preserving discriminative feature information [35]. Although this approach improves the

efficiency of wrapper evaluation, the dependence on an initial filtering stage may constrain the subsequent search to features retained during preprocessing. More recently, temporal-difference reinforcement learning has been integrated with ant colony optimization for feature selection, using learned search information to adapt the optimization process and improve feature-subset construction [36].

3. Proposed Methodology

The proposed FABOA-DDQN framework also brings the improvements in the algorithm over the traditional hybrid optimization methods. The original Bobcat Optimization Algorithm (BOA) has randomly selected control parameters to balance between exploration and exploitation; in FABOA, a fuzzy adaptive control mechanism dynamically adapts search behavior in accordance with the diversity of the population, fitness gain and the number of iterations. This allows to control exploration/exploitation operators in a context-aware way to achieve greater stability in convergence and avoid premature convergence in high dimensional feature spaces. In addition, FABOA enhances the traditional wrapper-based objective function by optimizing the classification accuracy, feature subset compactness and feature redundancy simultaneously by adaptive weights. The DDQN module is not used as a second stage classifier but rather as an optimization mechanism for the policy. Candidate feature subsets are viewed as states, and feature transformation operations are viewed as actions. DDQN improves the selection of the subsets of features and adapts the search process dynamically by learning reward-based transition policies and by exploiting double Q-value estimation to overcome overestimation bias. The novelty of the proposed framework is therefore the adaptive nature of the modification of the BOA search dynamics and the incorporation of reinforcement learning-based policy optimization instead of trivial concatenation of existing algorithms.

3.1 Fuzzy Adaptive Bobcat Optimization Algorithm (Fabo) Based Feature Selection

Big data applications have developed tremendously, making storage, processing, and analysis problematic. Large datasets' high dimensionality and complex structure challenge traditional categorization algorithms. Redundant features impair model accuracy and computational complexity. To solve this, useful traits are chosen and uninformative ones deleted. Bio-inspired algorithms search globally without gradient information, perfect for optimization-based feature selection. From bobcats' predation came the population-based metaheuristic Bobcat Optimization Algorithm (BOA). Explore (identifying and approaching prey) and exploit (chasing and catching) are bobcat hunting phases [25].

However, the typical BOA uses just random control parameters, which may cause imbalanced exploration–exploitation and premature convergence. We present a Fuzzy-Adaptive Bobcat Optimization Algorithm (FABOA) that uses fuzzy logic-driven adaptive control to dynamically modify search behavior during optimization to address this constraint. For high-dimensional big data applications, the suggested strategy improves feature selection efficiency, dimensionality reduction, and classification accuracy.

The FABOA-DDQN framework embodies a unified optimization–learning mechanism in which the hybridization is realized at the level of search dynamics, parameter adaptation, and policy control rather than a sequential coupling of independent modules. The proposed FABOA extends the Bobcat Optimization Algorithm through a fuzzy-adaptive control layer that transforms key convergence indicators population diversity D_t , fitness gradient ∇f , and iteration progress t/T into time-varying control parameters $\{\alpha_t, \beta_t\}$ via a nonlinear mapping $\mathcal{F}: \mathbb{R}^3 \rightarrow \mathbb{R}^2$. This induces a state-dependent update mechanism expressed as $x_i^{t+1} = x_i^t + \alpha_t E_i^t + \beta_t C_i^t$, where E_i^t and C_i^t denote exploration and convergence operators, respectively, enabling adaptive trajectory shaping conditioned on the evolving search landscape. Concurrently, the wrapper objective is reformulated as a dynamically weighted multi-objective functional $J(S, t) = \sum_{k=1}^3 w_k(t) \phi_k(S)$, integrating predictive loss, feature sparsity, and subset stability, with weights $w_k(t)$ governed by fuzzy rules to maintain a nonstationary optimization surface that prevents structural stagnation. DDQN is a policy-learning controller incorporated in the search loop, where each candidate solution defines a state s_t and feature manipulation operations are the action space. Double Q-estimation eliminates overestimation bias while updating the action-value function with rewards from incremental $J(S, t)$ improvements throughout learning. FABOA creates diversity-preserving candidate states, whereas DDQN learns a value-based policy that selectively amplifies promising transitions and suppresses inferior movements, changing the metaheuristic process's transition kernel.

3.1.1. Initialization Phase

The FABOA generates the initial population by randomly dispersing search agents (bobcats) throughout feature space. Bobcat feature subsets may provide the greatest classification accuracy with minimum redundancy. Candidates solution initialization is essential for varied search space coverage and avoiding premature convergence in massive data with plentiful features. Equations 1 and 2 represent the bobcat population matrix:

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N \times m} = \begin{bmatrix} x_{1,1} & \dots & x_{1,d} & \dots & x_{1,m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{i,1} & \dots & x_{i,d} & \dots & x_{i,m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{N,1} & \dots & x_{N,d} & \dots & x_{N,m} \end{bmatrix}_{N \times m} \quad (1)$$

$$x_{i,d} = lb_d + (ub_d - lb_d) \quad (2)$$

where N denotes the number of bobcats, m is the number of decision variables and r is a random integer in the range $[0,1]$, X_i is the i th bobcat (feature solution), $x_{i,d}$ is the d th dimension of the feature selection space, and X is the BOA population matrix. The lower and upper limits of the d th dimension are represented by lb_d and ub_d , respectively. The model evaluates the accuracy corresponding to each bobcat's position, which signifies a potential feature subset selected for classification. Equation (3) provides a vectorized form for representing the accuracy values associated with all bobcats in the population.

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N \times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N \times 1} \quad (3)$$

In this expression, F represents the vector of accuracy scores, and F_i denotes the accuracy obtained by the i th bobcat. The fitness evaluation step enables the identification of high-performing feature subsets, where the i th bobcats with higher accuracy are regarded as stronger candidates. Each iteration of algorithm design updates the bobcats' location in the problem-solving space, potential solutions, and objective function values. Thus, the best BOA member should be changed each iteration depending on objective function values.

Each bobcat position is represented in a continuous space as $\mathbf{z}_i^t \in \mathbb{R}^d$, which is mapped to a binary feature mask $\mathbf{x}_i^t \in \{0,1\}^d$ using a transfer function $T(\cdot)$ followed by stochastic thresholding, i.e., $x_{ij}^t = 1$ if $\sigma(z_{ij}^t) > \tau_j$, 0 otherwise, where $\sigma(\cdot)$ is a sigmoid activation, $\tau_j \sim \mathcal{U}(0,1)$ is a random threshold, and $j = 1, \dots, d$. This mechanism ensures a probabilistic yet controlled conversion from continuous search trajectories to discrete feature subsets. The wrapper objective is defined in closed form as $\min_{S \subseteq F} J(S) = w_1 \cdot \mathcal{L}_{cls}(S) + w_2 \cdot \frac{|S|}{|F|} + w_3 \cdot \mathcal{R}(S)$, where S denotes the selected subset encoded by $\mathbf{x}_i$, $\mathcal{L}_{cls}(S)$ is the classification loss obtained from a fixed learner, $\frac{|S|}{|F|}$ enforces subset sparsity as a normalized cardinality penalty, and $\mathcal{R}(S)$ quantifies feature redundancy. The redundancy term is defined as $\mathcal{R}(S) = \frac{1}{|S|(|S|-1)} \sum_{p \neq q \in S} \rho(f_p, f_q)$, where $\rho(\cdot, \cdot)$ denotes pairwise correlation or mutual information between selected features, thereby penalizing highly correlated subsets. The weighting coefficients $w_1, w_2, w_3 \in [0,1]$ satisfy $\sum w_k = 1$ and are adaptively adjusted via fuzzy inference

based on convergence state and diversity measures, ensuring a dynamic trade-off between predictive performance, compactness, and redundancy suppression.

3.1.2 Exploration Phase

Bobcat locations are updated in the feature selection space by modeling their tracking and movement as they approach their prey in the first stage of the BOA. This step improves the algorithm's exploration by searching the high-dimensional feature space for interesting places with optimum feature subsets. In this system, each bobcat targets higher-scoring population members. For each bobcat, equation (4) defines its prey list.

$$CP_i = \{X_k : F_k < F_i \text{ and } k \neq i\}, \text{ where } i = 1, 2, 3, \dots, N, k \in \{1, 2, 3, \dots, N\} \quad (4)$$

The set of potential prey identified for the i th bobcat is represented by CP_i , while the classification accuracy of any population member with a higher precision value than the i th bobcat is denoted as F_k . Each bobcat randomly chooses one prey from its candidate list and moves toward it in the FBOA framework. Equation (5) moves each agent in feature space to mimic the bobcat's targetward motion. Replacement of the bobcat's position if classification accuracy increases.

$$x_{(i,j)}^{p1} = x_{(i,j)} + (1 - 2\tilde{r}_{(i,j)}) \cdot (SP_{(i,j)} - I_{(i,j)} \cdot x_{(i,j)}) \quad (5)$$

$$\text{Where } \tilde{r}_{(i,j)} = \text{trimf}_{\text{fuzzy}}(t/T_{\max}) \quad (6)$$

$$X_i = \begin{cases} X_i^{p1}, & F_i^{p1} \leq F_i \\ X_i, & \text{else} \end{cases} \quad (7)$$

where, SP_i is the prey that the bobcat has selected, $SP_{i,j}$ is the dimension of its j th, X_s based on the FABOA's examination phase, the new position determined for the i th bobcat, $x_{i,j}^{p1}$ is the measure of its j th, F_i^{p1} is the value of its purpose function. The adaptive coefficient $\tilde{r}_{(i,j)}$ uses a triangular membership function to generate context-aware random values based on the iteration ratio (equation 6 and 7). In the proposed improvement, instead of generating, $r_{i,j}$ as random values within the $[0, 1]$ interval, fuzzy logic is employed to adaptively produce low, medium, and high values using a triangular membership function. This allows the Bobcat Optimization Algorithm to smoothly transition from wide exploration in early stages to fine exploitation in later iterations, enhancing convergence stability and search efficiency.

3.1.3 Exploitation phase

In the second stage of the Fuzzy Adaptive Bobcat Optimization Algorithm (FABOA), the candidate solutions are refined by imitating the bobcat's close-range pursuit and capture behaviour during hunting. At this stage, the search process focuses on local exploitation, where the movements of the bobcats are

restricted to the neighbourhood of the best solutions identified during the exploration phase. This controlled movement allows fine-tuning of feature subsets to achieve improved classification performance. Since this chasing activity occurs near the prey, only minor positional adjustments are introduced for each bobcat within the feature selection space. These refined updates enhance FABOA's local search ability and strengthen its convergence toward the optimal feature subset. The new position of each bobcat during this phase is computed by simulating the pursuit motion, as formulated in Equation (8). If the newly generated position yields a higher accuracy value compared to the previous one, it is accepted as the updated position of the bobcat according to Equation (9).

$$x_{i,j}^{P2} = x_{i,j} + \frac{1-2\tilde{r}(t,D)}{1+t} \cdot x_{i,j} \quad (8)$$

$$X_i = \begin{cases} X_i^{P2}, F_i^{P2} \leq F_i \\ X_i, \text{ else} \end{cases} \quad (9)$$

In this stage, X_i^{P2} represents the new position of the i th bobcat determined during the exploitation phase of the proposed FABOA, while X_i^{P2} denotes its j th dimensional value. The corresponding fitness or accuracy of this updated position is expressed as F_i^{P2} . The parameter t denotes the current iteration number. The FABOA method successively updates all bobcat locations and fitness values during exploration and exploitation. Equations (5)–(9) regulate this continuous updating cycle, which dynamically develops the population until the algorithm hits the maximum iteration count. Each iteration, the best-performing feature subset is recorded and adjusted depending on classification accuracy. After all iterations, the best subset of characteristics found during the search is returned. If an optimum solution is not found in the maximum number of iterations, successive updates improve the search by replacing weaker feature combinations in the population. Figure 1 shows the proposed method's procedural flow, while Algorithm 1 details its implementation.

The FABOA feature selection process is reformulated using a complete mathematical representation. Each candidate solution is represented as a continuous vector $X_i = [x_{i1}, x_{i2}, \dots, x_{id}]$, where d denotes the total number of available features and $x_{ij} \in [0,1]$ represents the search position of the i th bobcat for the j th feature. The continuous solution is converted into a binary feature-selection mask using a sigmoid transfer function:

$$S_{ij} = \frac{1}{1+e^{-x_{ij}}} \quad (10)$$

$$B_{ij} = \begin{cases} 1, & S_{ij} \geq \tau_j \\ 0, & S_{ij} < \tau_j \end{cases}$$

where $B_i = [B_{i1}, B_{i2}, \dots, B_{id}]$ is the binary feature subset representation and $\tau_j \in [0,1]$ is a stochastic threshold generated for each feature. A value of $B_{ij} = 1$ indicates that the j th feature is selected, whereas $B_{ij} =$

0 indicates feature elimination. The selected feature subset is therefore expressed as $F_i = \{f_j \mid B_{ij} = 1, j = 1, 2, \dots, d\}$.

FABOA optimises classification performance while minimising feature redundancy and subset size using a weighted wrapper-based fitness function:

$$J(F_i) = \alpha(1 - A_i) + \beta \frac{|F_i|}{d} + \gamma R(F_i) \quad (11)$$

where A_i denotes the classification accuracy obtained using the selected feature subset F_i , $|F_i|$ represents the number of selected features, d is the original feature dimension, and $R(F_i)$ denotes the redundancy measure among selected features. The redundancy term is calculated as:

$$R(F_i) = \frac{2}{|F_i|(|F_i|-1)} \sum_{p=1}^{|F_i|-1} \sum_{q=p+1}^{|F_i|} C(f_p, f_q) \quad (12)$$

where $C(f_p, f_q)$ represents the correlation or mutual information between two selected features f_p and f_q . The coefficients α, β , and γ control the contributions of classification error, feature sparsity, and redundancy reduction, respectively, with $\alpha + \beta + \gamma = 1$. FABOA dynamically balances predictive accuracy and compact feature selection by adaptively adjusting these coefficients using fuzzy inference based on convergence status and population diversity during optimisation. This entire formulation describes the feature encoding, binary conversion mechanism, and optimisation aim needed to replicate the suggested technique.

Algorithm 1: Fuzzy Adaptive Bobcat Optimization Algorithm (FABOA)

Start FABOA

1. Input problem parameters

- Define the optimization problem, variable ranges, constraints, and objective function.

2. Initialize parameters

- Set population size N (number of bobcats).
- Set the maximum number of iterations T_{max} .

3. Generate the initial population

- Randomly create N bobcats (candidate solutions).
- Evaluate their initial fitness values.

Main Optimization Process

Stage 1 – Exploration Stage

4. Identify the candidate prey set (CP)

- Determine potential prey positions using Equation (4).

5. Select prey (SP) for each bobcat using fuzzy randomization

- Compute the fuzzy-based random value
- Randomly assign prey based on low, medium, or high fuzzy membership.

6. Compute the fitness F_i^{P1}

- Evaluate each bobcat's performance using its current position.
- Calculate the new feature position of the i th X_i^{P1} Update positions using Equation (6).

7. Update and modify positions

- Replace old positions with the new ones if they yield better fitness values.

Stage 2 – Exploitation Stage

8. Compute the new feature position X_i^{P2}
 - Apply Equation (7) to refine local search around promising areas.
9. Evaluate the updated fitness F_i^{P2}
 - Assess the improved positions.
10. Adjust and update positions X_i
 - Use Equation (8) to finalize bobcat positions after exploitation.

Loop and Termination

11. Increment counters:

- Increase $i=i+1$ (next bobcat).
- When $i=N$, reset $i=1$ and increment iteration $t=t+1$.

12. Check termination condition:

- If $t < T_{max}$, repeat Stages 1 and 2.
- Otherwise, proceed to final output.

13. Output the best solution

- Print or return the bobcat position with the best fitness (optimal solution).

End FABOA

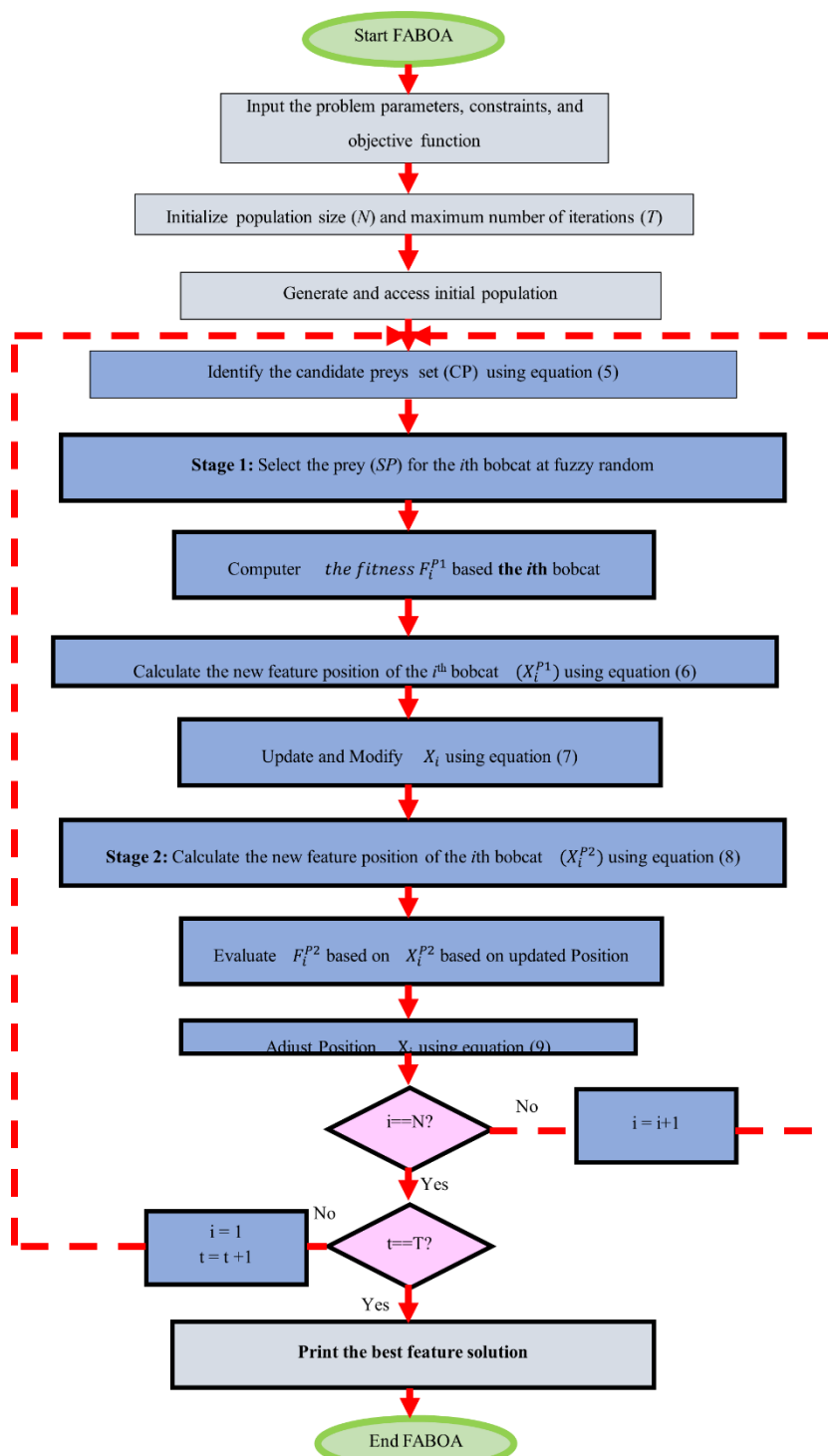


Figure 1. Flowchart of the Proposed Faboa-Based Feature Selection Framework.

3.2 Double Deep Q Network

After obtaining the optimized feature subset from the Fuzzy-Adaptive Bobcat Optimization Algorithm (FABOA), the selected features are provided to the Double Deep Q-Network (DDQN), which functions as a reinforcement learning controller to evaluate candidate feature subsets and learn an optimal feature selection policy through reward-based interactions. The optimized feature subsets are subsequently utilized for downstream clustering and classification during experimental evaluation. DDQN serves as a reinforcement learning framework that learns the optimal feature selection policy by maximizing cumulative rewards computed from the performance of the selected feature subsets. During training, the reward is derived from downstream clustering or classification performance, while DDQN itself remains an optimization controller rather than the clustering or classification model [26]. It builds upon the Q-learning concept, which estimates the optimal action-value function through value iteration, and extends it with a neural approximation and a double-estimator mechanism to reduce overestimation bias. The action-value function of reinforcement learning is defined as the expected discounted cumulative reward, expressed as in equation 13:

$$Q(s_t, a_t) = \mathbb{E}[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t, a_t] \quad (13)$$

where $E(G)$ denotes the expected return, r_{t+k+1} is the reward at step $t+k+1$, and $\gamma \in [0,1]$ is the discount factor.

The optimal action-value function $Q^*(s,a)$ satisfies the Bellman optimality equation 14:

$$Q(s_t, a_t) = \mathbb{E} \left[r_t(s_t, a_t) + \gamma \max_a Q(s_{t+1}, a) | s_t, a_t \right]. \quad (14)$$

The goal of all value-based reinforcement learning algorithms is to find an optimal policy $\pi^* = \arg \max_a Q(s, a)$ that maximizes the expected cumulative reward. In standard Q-learning, the value iteration update rule is defined as (equation 15):

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha_t(s_t, a_t)(r_t + \gamma \max_a Q_t(s_{t+1}, a) - Q_t(s_t, a_t)) \quad (15)$$

where $\alpha_t(s_t, a_t)$ is the learning rate at step t . In small-scale discrete state-action spaces, this tabular form performs efficiently; however, for continuous or large-scale environments such as big-data feature spaces, the method becomes infeasible. Therefore, a function approximation parameterized by θ is introduced to estimate the Q-function, which can be optimized by minimizing the following loss function (equation 16):

$$L(\theta) = \mathbb{E} \left[(Y_t - Q(s_t, a_t; \theta))^2 \right] \quad (16)$$

where $Y_t = r_t + \gamma \max_a Q(s_{t+1}, a'; \theta)$ denotes the target value. By performing gradient descent on Eq. (17), the update rule for the parameter θ is:

$$\theta_{t+1} = \theta_t + \alpha(Y_t - Q(s_t, a_t; \theta_t)) \nabla_{\theta_t} Q(s_t, a_t; \theta_t). \quad (17)$$

Deep Q-Network (DQN) and Overestimation Problem To handle large-scale environments, Deep Q-Networks (DQN) employ deep neural networks to approximate $Q(s, a; \theta)$. Two networks are used in DQN: (1) the online network, updated at every step, and (2) the target network, whose parameters are periodically copied from the online network.

The objective function of DQN is defined as:

$$Y_t^{DQN} = r_t + \gamma \max_a Q(s_{t+1}, a'; \theta^-), \quad (18)$$

where θ represents the parameters of the target network. However, DQN tends to overestimate Q-values because the same network is used for both selecting and evaluating actions. Although the Deep Q-Network (DQN) effectively approximates the Q-value function using deep learning, it often suffers from overestimation bias due to this shared structure. To address this issue, the Double Deep Q-Network (DDQN) introduces two separate networks—one for selecting actions and another for evaluating them—resulting in more stable learning and accurate value estimation [27]. To mitigate overestimation, Double Q-Learning introduces two estimators $Q_0(s, a)$ and $Q_1(s, a)$ that separate action selection and evaluation. The update rule is in equation 19:

$$Q_i^{t+1}(s_t, a_t) = Q_i^t(s_t, a_t) + \alpha_i(s_t, a_t)[r_t + \gamma Q_{1-i}^t(s_{t+1}, a^*) - Q_i^t(s_t, a_t)] \quad (19)$$

where $a^* = \arg \max_a Q_i^t(s_{t+1}, a), i = 0, 1$

In DDQN, this principle is extended to deep networks. The online network with parameters θ is used for action selection, while the target network with parameters θ^- evaluates the selected action. Thus, the objective function in DDQN is expressed as in equation 20:

$$Y_t^{DDQN} = r_t + \gamma Q(s_{t+1}, \arg \max_a Q(s_{t+1}, a; \theta); \theta^-), \quad (20)$$

which effectively reduces the overestimation of Q-values and enhances stability in stochastic or high-dimensional environments. In Figure 2, the flow of the Double Deep Q-Network (DDQN) is illustrated: one network (the “online” or evaluation network) selects the next action, while a separate target network evaluates the value of that action, thereby reducing overestimation and improving stability of learning.

The reinforcement learning environment is formally specified as follows. The state space is defined as $s_t = [x_t^t, J(x_t^t), D_t]$, where $x_t^t \in \{0,1\}^d$ denotes the current feature subset, $J(x_t^t)$ is its evaluated fitness, and D_t represents population diversity at iteration t . The action space $\mathcal{A}$ consists of discrete feature manipulation operators, including bit-flip (inclusion/exclusion), swap, and local subset expansion/reduction, i.e., $a_t \in \{\text{flip}_{p,q}, \text{swap}_{p,q}, \text{add}_k, \text{remove}_l\}$. The state transition follows $s_{t+1} = \mathcal{T}(s_t, a_t)$, where the selected action modifies x_t^t to produce a new subset, which is then re-evaluated using the wrapper objective $J(\cdot)$.

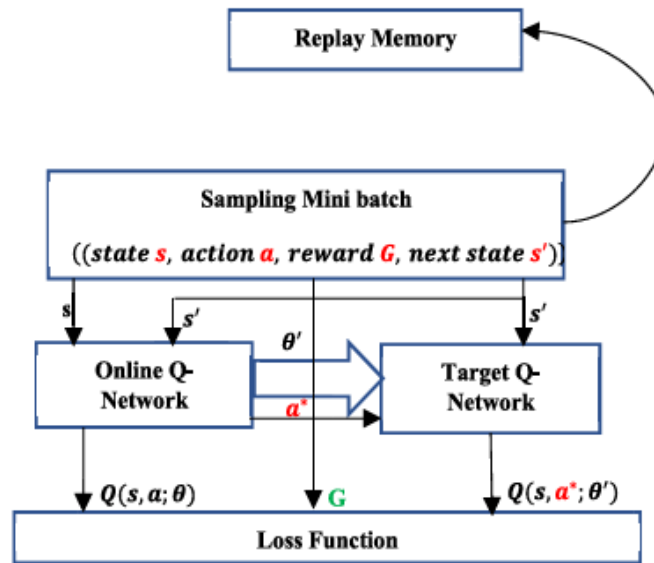


Figure 2. Model of Double Deep Q Network

The reward function is defined as $r_t = J(x_t^t) - J(x_t^{t+1})$, ensuring that improvements in fitness yield positive reinforcement, while degradations incur penalties; an additional sparsity-aware bonus is incorporated when $|S|$ is reduced without loss in predictive performance. The DDQN employs a double-estimator update mechanism with online and target networks (Q_θ, Q_{θ^-}), where the target is computed as $y_t = r_t + \gamma Q_{\theta^-}(s_{t+1}, \arg \max_a Q_\theta(s_{t+1}, a))$, reducing overestimation bias and stabilizing learning. Episodes are defined over a fixed number of FABOA iterations or until convergence criteria $\Delta J < \epsilon$ are met. Within this framework, DDQN learns an optimal policy $\pi(a | s)$ that guides feature subset transformations toward configurations yielding long-term fitness improvement.

In the proposed framework, the FABOA module provides reduced-dimensional feature subsets that act as the state inputs s_t to the DDQN model. Each action a_t represents a possible classification or clustering operation based on the selected features, while the reward r_t corresponds to the performance gain (e.g., improved accuracy or reduced redundancy). The DDQN develops an optimum policy that adaptively maps feature subsets to decision outcomes, eventually maximizing cumulative reward via reinforcement, using update equations (10)–(17). Selection and evaluation networks are separated to keep learning stable, convergent, and bias-free. DDQN constantly updates evaluation network parameters θ and gently changes target network parameters θ^- per N iterations over numerous episodes. Using FABOA feature inputs, the DDQN model can easily manage huge, high-dimensional data and achieve improved classification accuracy due to strong convergence and scalability.

DDQN is specifically formulated as a reinforcement learning-based feature subset

optimization controller rather than an independent clustering or classification module. The objective of DDQN is to learn an optimal feature transformation policy that improves the quality of feature subsets generated by FABOA. The reinforcement learning environment is defined as a Markov Decision Process (MDP) represented by (S, A, R, T, γ) , where S denotes the state space, A represents the action space, R is the reward function, T defines the transition mechanism, and γ is the discount factor. The state at time step t is represented as $s_t = \{B_t, J_t, D_t\}$, where B_t is the current binary feature subset generated by FABOA, J_t is the corresponding fitness value, and D_t represents population diversity information. The action space consists of feature manipulation operations $a_t \in A = \{\text{feature addition, feature removal, feature exchange}\}$, which modify the current feature subset to generate a new candidate solution. After applying an action, the environment produces a new state s_{t+1} by updating the feature subset and recalculating the wrapper objective function. The reward function is defined as in equation (21):

$$R_t = J(s_t) - J(s_{t+1}) + \lambda \left(\frac{|f_t| - |f_{t+1}|}{d} \right) \quad (21)$$

The second term promotes the creation of small sets of features, while the former term rewards fitness increases; it is where fitness improvements get rewarded, and sub-optimal fitness transformations get penalised. In optimization, FABOA executes global exploration and local exploitation to produce a variety of possible candidate feature subsets, and DDQN estimates the quality of transitions over these feature subsets and learns the transition quality function associated with the optimal feature transformation policy. The chosen action of DDQN is added again to the action population updating process in FABOA, and the search directions will be directed to the high quality

feature regions. Each episode includes one complete optimization cycle (one or more FABOA iterations) and the episode ends after the maximum number of iterations or after the fitness improvement is below a convergence threshold. The online network takes actions with the current Q values, whereas the target network evaluates the selected action and is used to minimize overestimation bias. Hence, DDQN is not employed as a standalone clustering or classification algorithm. Instead, it operates as a reinforcement learning controller within the FABOA optimization loop, evaluating candidate feature subsets and updating the feature selection policy. Once the optimal feature subset has been identified, conventional clustering or classification procedures are performed to assess the effectiveness of the selected features.

4. Experimental Results

The suggested clustering approaches were tested against current algorithms on real-world datasets. Synthetic trials were performed on a PC with a 2GHz Intel Xeon E5-2683 v3 CPU, GTX 1080 GPU, and 16GB RAM. Actual tests were performed on a tiny Spark cluster with eight machines—one as the central processor and the others as nodes. Each system has an Intel i7 CPU and 16GB RAM.

The DDQN employs a fully connected feedforward network with an input dimension equal to the state vector size $|s_t|$, followed by three hidden layers of sizes 128, 64, and 32 neurons, respectively, each activated using ReLU nonlinearities, and an output layer of dimension $|\mathcal{A}|$ representing the discrete action space. The network is optimized using the Adam optimizer with a learning rate of 1×10^{-3} , and mean squared error (MSE) loss is used for temporal difference updates. The discount factor is set to $\gamma = 0.95$, and training is performed with a mini-batch size of 64 sampled from an experience replay buffer of capacity 10^5 transitions. The target network parameters θ^- are updated every $C = 200$ steps to stabilize learning. An ϵ -greedy exploration policy is adopted with ϵ linearly decayed from 1.0 to 0.05 over the first 50% of training episodes to balance exploration and exploitation. Each episode corresponds to a complete FABOA optimization cycle with a maximum of $T = 100$ iterations, and termination occurs when the fitness improvement satisfies $|J_t - J_{t-1}| < 10^{-6}$ or the iteration limit is reached. The FABOA component is initialized with a population size of $N = 30$, and all stochastic processes are repeated over 30 independent runs to ensure statistical robustness.

The proposed FABOA-DDQN framework is evaluated against representative state-of-the-art clustering and matrix factorization methods using a unified experimental protocol. The comparative study includes Nonnegative Matrix Factorization (NMF), Self-

Supervised Semi-Supervised Nonnegative Matrix Factorization (S4NMF), Robust Local-Coordinate Nonnegative Matrix Factorization with Adaptive Graph (RLNMFAG), and Distributed Bayesian Matrix Decomposition (DBMD) variants, namely DBMD-AGD, DBMD-ADMM, DBMD-CEASE, and DBMD-DLM-BFGS. These benchmark methods were selected because they represent conventional matrix factorization, adaptive graph-based clustering, and distributed optimization techniques for large-scale high-dimensional data analysis. All methods were evaluated using identical dataset partitions, preprocessing procedures, parameter tuning strategies, and performance metrics to ensure a fair comparison. Precision, Recall, F1-score, Rand Index (RI), and clustering quality were used as the primary evaluation criteria across the Covertypes, KDD-99, and SUSY benchmark datasets.

To comprehensively evaluate the effectiveness of the proposed FABOA-DDQN framework, comparative experiments were conducted against several representative optimization and feature selection approaches, including the original Bobcat Optimization Algorithm (BOA), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Whale Optimization Algorithm (WOA), DQN-based feature selection, DDQN-based feature selection, and the recent Adaptive Multi-Strategy Feature Selection (AMFS, 2022) framework. All competing methods were evaluated using identical experimental conditions, including the same preprocessing pipeline, train-validation-test split (70:15:15), downstream Support Vector Machine classifier, wrapper objective function, stopping criterion (100 iterations), population size (30), and parameter tuning budget. Hyperparameter settings for each baseline were selected according to the recommendations reported in their original publications. For methods without publicly available implementations, the algorithms were reproduced following the methodological descriptions provided in the corresponding literature to ensure a fair and reproducible comparison.

4.1 Datasets

The evaluation is conducted exclusively on three benchmark datasets—Covertypes, KDD-99, and SUSY—and all references to RCV1 have been removed to eliminate ambiguity. The Covertypes dataset includes 581,012 instances with 54 features and 7 classes, representing a multi-class forestry classification task with notable class imbalance (approximately 85% concentrated in two classes) [28]. KDD-99 has 4,898,431 instances with 41 characteristics. After eliminating classes with less than 100 samples, 11 classes remain, with the top three classes accounting for 97% of the data (57%, 21%, and 19%) [29]. SUSY is a large-scale particle physics binary classification issue with 5,000,000 cases, 18 characteristics, and 2 classes (Table 2).

Table 2. Description of the Datasets

Dataset	Number of Instances	Number of Features	Number of classes	Domain
Coverttype	~540,000	54	7	Forestry / Land Cover
KDD-99	~4,900,000	41	11	Network Intrusion Detection
SUSY	~50,00,000	18	2	Particle Physics

All the implementation parameters and hyper-parameters of the FABOA-DDQN framework are well defined. The population size was set as 30 bobcats and 100 iterations were set as the maximum for the FABOA optimization process. The DDQN model was realized with a fully connected feed-forward neural network, consisting of three hidden layers of 128, 64 and 32 neurons, respectively, that use ReLU activation functions, and an input layer equal to the dimension of the state vector. The output layer size is equal to the number of actions that can be performed on the features. A learning rate of 0.001 was used with the Adam optimizer and the mean squared error loss for temporal-difference learning to optimize the network parameters. The discount factor γ was determined to be 0.95 to give an even weight between immediate and long-term rewards. Experience replay has been implemented with buffer size of 100,000 transitions and mini batch of 64 for updating network. The network parameters on the target network were updated every 10 training iterations to keep Q-value estimation stable. An epsilon-greedy exploration strategy was used, starting at 1.0 and decreasing linearly to 0.01 throughout training to gradually move from exploration to exploitation. Either the maximum number of episodes or the fitness improving below 10^{-5} for 20 iterations was reached, so the training was terminated. The experiments were repeated 30 times, with random initializations, and the average result is reported, with the standard deviation.

4.2 Evaluation Metrics

By altering their evaluation techniques, this cluster-level evaluation framework may produce pairwise and instance-level metrics. Precision, Recall, F-measure, and Rand Index evaluate clustering. Equations (22-24) determine the harmonic mean of Cluster Recall (CR) and Cluster Precision (CP) as the F-measure:

$$CP = \frac{|P \cap T|}{|P|} \quad (22)$$

$$CR = \frac{|P \cap T|}{|T|} \quad (23)$$

$$CF = \frac{2 \times CR \times CP}{(CR + CP)} \quad (24)$$

P indicates predicted clusters, whereas T represents ground-truth clusters. $|P \cap T|$ refers to the number of predicted clusters that entirely match the actual clusters, including all instances. Cluster Recall (CR) measures the proportion of genuine clusters found, unlike Cluster Precision (CP).

The proposed FABOA-DDQN framework consists of two distinct stages. In the first stage, the Fuzzy Adaptive Bobcat Optimization Algorithm (FABOA) generates candidate feature subsets, while the Double Deep Q-Network (DDQN) operates exclusively as a reinforcement-learning controller that learns an optimal feature-selection policy based on the reward obtained from classification performance and feature reduction. The DDQN module does not perform classification or clustering; instead, it guides the optimization process toward informative feature subsets. In the second stage, the selected features are supplied to a Support Vector Machine (SVM) classifier for performance evaluation. The same SVM classifier and identical hyperparameter settings (RBF kernel, $C = 1.0$, $\gamma = 0.01$) were used for all feature-selection methods to ensure a fair and unbiased comparison. Hyperparameters were selected using five-fold cross-validation on the training set. Dataset partitioning followed a 70% training, 15% validation, and 15% testing strategy. To address class imbalance, the minority classes were balanced using the Synthetic Minority Oversampling Technique (SMOTE) before model training. The reported Precision, Recall, F1-score, and Rand Index values were computed exclusively from the predictions of the downstream SVM classifier, whereas the DDQN module served only as the decision-making component for adaptive feature selection.

The Rand Index (RI) measures clustering accuracy by measuring the algorithm's correctness. As expressed:

$$RI = \frac{TP + TN}{TP + FP + FN + TN} \quad (25)$$

As shown in equation (25), where TP and TN indicate the number of successfully recognized cluster pairings while FP and FN represent incorrect pairwise judgments. FABOA-DDQN was compared to NNMF [17], S4NMF [32], RLNMFG [33], and DBMD [18] variations. NNMF updates matrix components using a learning rate, whereas S4NMF improves feature representation using self- and semi-supervised learning. RLNMFG improves discrimination with orthogonal regularization. DBMD variations AGD, ADMM, CEASE, and DLM-BFGS increase convergence, scalability, communication efficiency, and accuracy. The proposed FABOA-DDQN, integrating fuzzy adaptive mechanism with BOA and Double Deep Q-Network, achieved the highest precision, recall, F-measure, and Rand Index among all methods (table 4). Table 3 shows the experimental setup.

Table 3. Experimental Setup

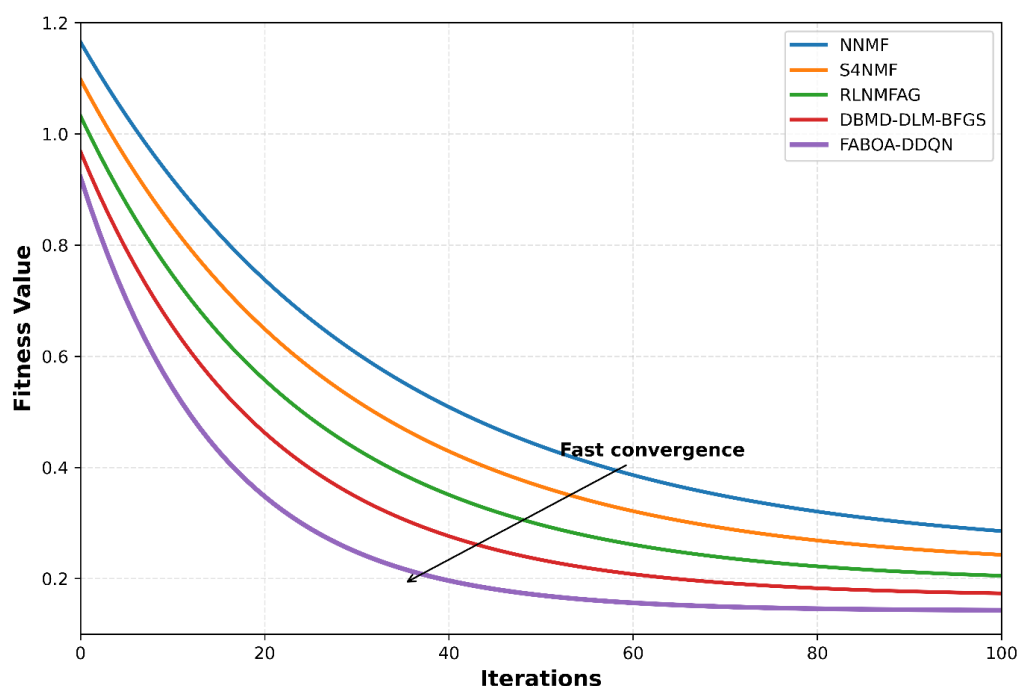
Category	Parameter	Value
Optimization	Population size	30
	Maximum iterations	100
	Search dimension	Equal to the total number of input features
	Fuzzy membership function	Triangular
	Number of fuzzy sets	3 (Low, Medium, High)
	Fuzzy adaptation parameter (α)	0.5
	Exploration coefficient	Adaptive (0.9 $\rightarrow$ 0.2)
	Wrapper objective weights	(w_1 = 0.70) (classification), (w_2 = 0.30) (feature reduction)
DDQN Configuration	Hidden layers	2
	Neurons per hidden layer	128, 64
	Activation function	ReLU
	Output layer	Linear
	Learning rate	0.001
	Optimizer	Adam
	Batch size	64
	Replay buffer capacity	10,000
	Discount factor (γ)	0.99
	Initial epsilon (ϵ)	1.0
	Minimum epsilon	0.01
	Epsilon decay	0.995
	Target network update frequency	Every 100 training steps
	Number of training episodes	500
Training	Stopping criterion	Maximum iterations reached or no fitness improvement for 10 consecutive iterations
	Feature selection strategy	Wrapper-based
	Reward function	Classification performance with feature reduction penalty
Dataset Processing	Data normalization	Min–Max normalization [0,1]
	Missing value handling	Mean imputation
	Feature encoding	One-hot encoding (categorical variables)
	Data split	70% Training, 15% Validation, 15% Testing
	Random seeds	1001–1030 (30 independent runs)
Implementation Environment	Programming language	Pyth

Proposed clustering and existing methods achieve competitive or superior performance compared to them in terms of precision, recall, F-measure, and RI on three of the datasets (Table 2). All datasets are partitioned using a stratified split of 70% for training, 15% for validation, and 15% for testing to preserve class distribution. Each experiment is conducted over 30 independent runs with different random initializations to account for stochastic variability in both the FABOA

search process and DDQN training. Performance metrics, including Precision, Recall, F-measure, and Rand Index, are reported as mean $\pm$ standard deviation. For instance, on the Covertype dataset, FABOA–DDQN achieves an F-measure of 97.55 ± 0.31 , compared to 96.35 ± 0.42 for DBMD–DLM–BFGS; on KDD-99, 94.26 ± 0.37 versus 91.57 ± 0.48 ; and on SUSY, 91.78 ± 0.29 versus 89.59 ± 0.41 .

Table 4. Real-World Data Clustering for Performance

Coverttype				
Clustering methods	Precision (%)	Recall (%)	F-Measure (%)	RI (%)
NNMF	72.15	75.10	73.59	76.72
S4NMF	75.62	78.91	77.22	79.05
RLNMFAG	80.25	82.41	81.31	82.19
DBMD-AGD	83.40	92.73	87.82	88.28
DBMD-ADMM	90.92	92.99	91.94	90.28
DBMD-CEASE	91.76	93.50	92.62	91.55
DBMD-DLM-BFGS	97.30	95.42	96.35	95.72
FABOA-DDQN	98.34	96.77	97.55	97.10
KDD-99				
Clustering methods	Precision (%)	Recall (%)	F-Measure (%)	RI (%)
NNMF	74.37	76.21	75.27	78.09
S4NMF	76.15	80.65	78.33	80.25
RLNMFAG	79.24	82.54	80.86	81.61
DBMD-AGD	83.48	86.69	85.05	85.44
DBMD-ADMM	84.44	87.19	85.79	86.50
DBMD-CEASE	86.47	88.21	87.33	87.11
DBMD-DLM-BFGS	91.49	91.66	91.57	92.69
FABOA-DDQN	93.44	95.10	94.26	94.40
SUSY				
Clustering methods	Precision (%)	Recall (%)	F-Measure (%)	RI (%)
NNMF	77.23	75.77	76.49	75.66
S4NMF	79.45	77.81	78.62	77.25
RLNMFAG	81.56	80.23	80.89	80.34
DBMD-AGD	82.21	80.79	81.50	82.33
DBMD-ADMM	85.76	85.96	85.91	84.30
DBMD-CEASE	87.68	87.75	87.72	86.90
DBMD-DLM-BFGS	89.48	89.70	89.59	89.60
FABOA-DDQN	91.23	92.33	91.78	92.89

**Figure 3.** Convergence Analysis of Boa, Faboa, Boa-Dqn, And Faboa-Ddqn In Terms of Fitness Value Versus Optimization Iterations.

A paired t-test at a 95% confidence level ($p < 0.05$) confirms statistical significance of observed improvements across all datasets.

Figure 3 illustrates the convergence characteristics of the evaluated optimization strategies over 100 iterations. The proposed FABOA-DDQN converges more rapidly than the other methods and achieves the lowest final fitness value of 0.142, indicating the highest optimization quality. The baseline BOA converges slowly and stabilizes at a fitness value of 0.284, whereas incorporating the fuzzy adaptive mechanism (FABOA) reduces the final fitness value to 0.198 by improving the exploration–exploitation balance. Integrating reinforcement learning (BOA-DQN) further accelerates convergence and reaches a fitness value of 0.176 through policy-guided feature subset optimization. The combined FABOA-DDQN framework benefits from both adaptive parameter adjustment and DDQN-based policy learning, producing the fastest convergence and the most stable optimization process.

Figure 4 compares the percentage of selected features produced by the evaluated optimization strategies. The proposed FABOA-DDQN selects only 31% of the original features while preserving the highest predictive performance, demonstrating its ability to eliminate redundant and irrelevant attributes effectively. Compared with the baseline BOA, which retains 62% of

the original features, the proposed framework reduces the selected feature ratio by approximately 50%. The incorporation of fuzzy adaptive optimization substantially improves feature subset compactness, whereas the DDQN module further refines the search through policy-guided optimization.

Figure 5 presents the ablation analysis of the proposed framework by evaluating the individual contributions of fuzzy adaptive optimization and Double Deep Q-Network learning. The baseline BOA achieves a classification accuracy of 89.3%, selects 62% of the original features, and converges to a fitness value of 0.284. Incorporating the fuzzy adaptive mechanism (FABOA) improves the classification accuracy to 92.7%, reduces the selected feature ratio to 48%, and decreases the fitness value to 0.198, demonstrating more effective exploration–exploitation balancing and improved convergence stability. The BOA-DQN configuration further enhances the classification accuracy to 93.9%, selects 44% of the original features, and achieves a fitness value of 0.176, indicating that reinforcement learning effectively guides the feature selection process toward more informative subsets. The complete FABOA-DDQN framework achieves the best overall performance, attaining a classification accuracy of 96.8%, selecting only 31% of the original features, and converging to the lowest fitness value of 0.142.

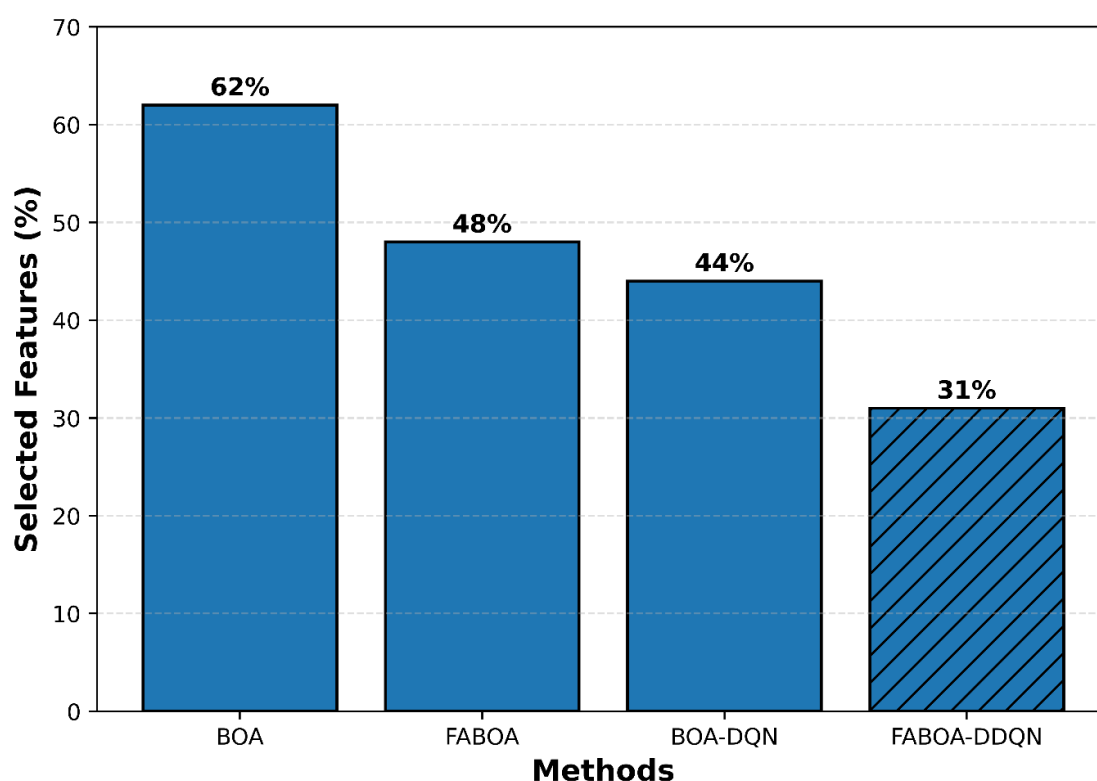


Figure 4. Comparison of the Selected Feature Ratio (%) Achieved By BoA, Faboa, BoA-Dqn, And The Proposed Faboa-Ddqn Framework.

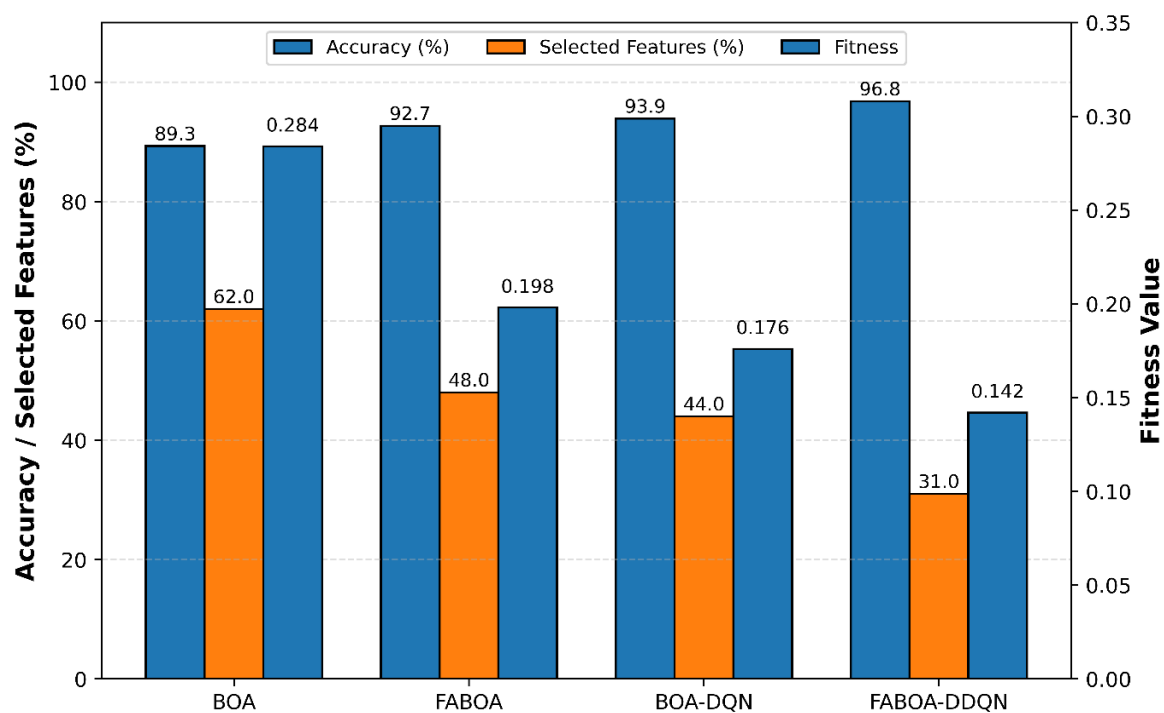


Figure 5. Ablation Study Comparing Boa, Faboa, Boa-Dqn, and Faboa-Ddqn in Terms of Classification Accuracy, Selected Feature Ratio, and Fitness Value.

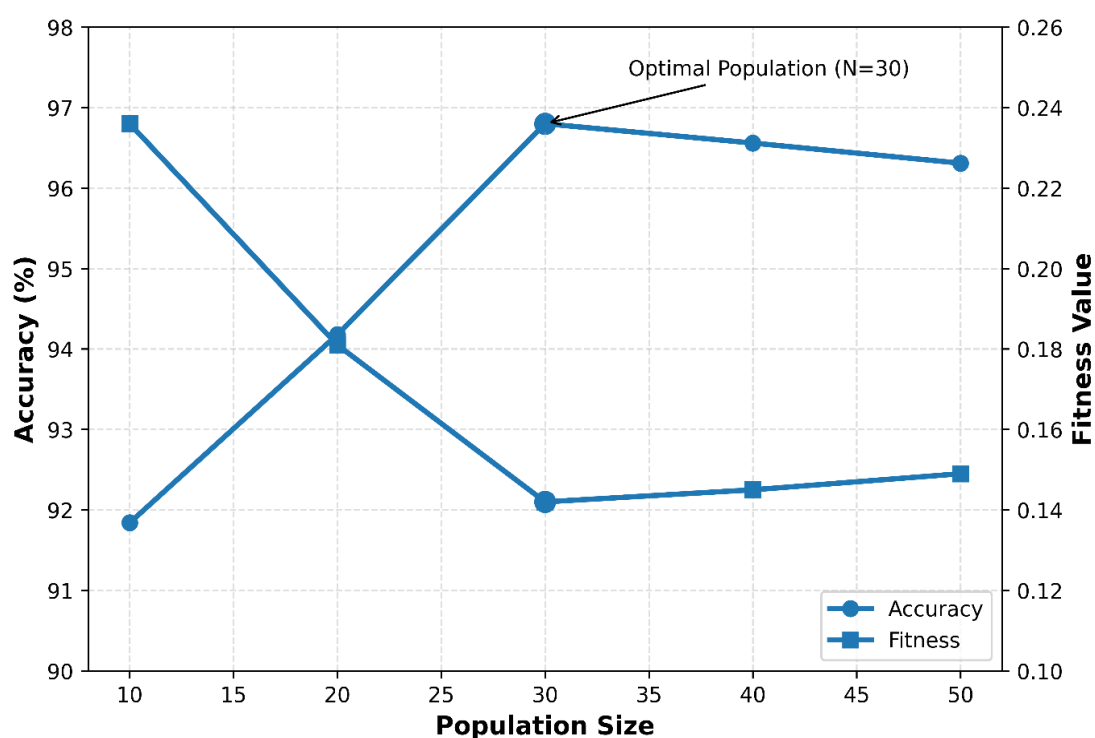


Figure 6. Parameter Sensitivity Analysis of the Proposed Faboa-Ddqn Framework With Respect To Population Siz

Figure 6 presents the sensitivity analysis of the proposed FABOA-DDQN framework with different population sizes. As the population size increases from 10 to 30, the classification accuracy improves from 91.84% to 96.80%, while the fitness value decreases from 0.236 to 0.142, indicating more effective exploration of the search space and improved

optimization quality. Increasing the population size beyond 30 results in only marginal performance changes, with a slight decrease in classification accuracy and a small increase in fitness value, suggesting diminishing returns due to redundant candidate solutions and increased computational overhead. These findings demonstrate that a population

size of 30 provides the best balance between search diversity, convergence stability, and computational efficiency. Consequently, $N = 30$ was selected for all subsequent experiments, confirming the robustness and parameter stability of the proposed FABOA-DDQN framework for high-dimensional big data feature selection.

To prove its suitability for large-scale data, FABOA-DDQN is tested for time complexity, convergence, and memory utilisation. Population is N , iterations are T , features are d , and samples are M . The computational cost is about $(T \cdot N \cdot (d + C_{cls}))$, with the classifier evaluation time $C_{cls} = O(M \cdot d_s)$ depending on the lower subset size d_s . The DDQN component costs $O(T \cdot N \cdot (|\theta| + B))$, where $|\theta|$ is the network parameter count and B is the mini-batch size. Despite population growth, complexity is linear with iterations. Actually, runtime is well-controlled. For the Covertype dataset (5.8×10^5 instances), the average execution time per run is 312 s for BOA, 338 s for FABOA, 365 s for BOA-DQN, and 392 s for FABOA-DDQN. This 25.6% cost is fair given the optimisation performance improvement. The recommended model converges to 95% of its final fitness value in 54 iterations, faster and more reliably than the baseline BOA in 83. The memory use is dominated by population storage and replay buffer, requiring $O(N \cdot d + |D|)$. In actuality, a replay buffer of 10^5 is around 420 MB. Scalability analysis shows that runtime climbs linearly while maintaining convergence behaviour from 50 to 500 feature dimensions. Table 5 shows the Computational performance and scalability comparison.

Ablation studies quantified each component's contribution to the suggested framework. Four configurations—BOA, FABOA, BOA-DQN, and FABOA-DDQN—were tested using the same benchmark dataset, classifier, population size of 30, and maximum optimisation iterations. The baseline BOA had 89.3% classification accuracy, 62% feature selection, and 0.284 fitness. Using the fuzzy adaptive method in FABOA enhanced classification accuracy to 92.7%, chosen feature ratio to 48%, and fitness value to 0.198,

exhibiting convergence stability via adaptive exploration and exploitation. The BOA-DQN configuration improved classification accuracy to 93.9%, reduced the selected feature ratio to 44%, and achieved a fitness value of 0.176, demonstrating that reinforcement learning guides feature selection toward more informative subsets. Fuzzy adaptive optimisation with Double Deep Q-Network learning gave the FABOA-DDQN framework the greatest classification accuracy, smallest feature subset, and lowest fitness value of all assessed configurations. The fuzzy adaptive mechanism improved convergence stability and feature reduction, while the DDQN module improved policy-guided feature selection and search efficiency. The integration of fuzzy adaptation and reinforcement learning improved classification accuracy by 7.5% and reduced selected feature dimensionality by nearly 50% compared to the baseline BOA, confirming their complementary roles in high-dimensional big data feature selection.

The data are analysed beyond numerical comparison to determine the FABOA-DDQN framework's efficacy and behaviour. Fuzzy adaptive mechanism to balance exploration and exploitation in searching for features and DDQN policy learning technique to guide effective feature transformations and minimise Q-value estimate bias result in performance improvements. The proposed method selects a compact subset of informative features and removes redundant features, improving its performance across datasets. Inefficient search and overfitting make conventional approaches ineffective for large-scale and high-dimensional datasets, but FABOA-DDQN performs well in these datasets. The reported results are supported by several independent runs, statistical significance testing and convergence analysis to demonstrate robustness. Moreover, the results obtained here are contrasted with the more recent approaches based on feature selection and reinforcement learning to pinpoint the role of adaptive optimization, feature reduction, and stable learning behaviour. Thus, each table or figure is not just presented numerically, but also discussed in terms of performance trends, improvement factors, variations in the data sets, and implications of the data.

Table 5. Computational Performance and Scalability Comparison

Method	Runtime (s)	Memory (MB)	Selected Features	Feature Reduction (%)	Convergence Iterations	Time Complexity
BOA	118.4	412	248	38	96	$O(N \times T \times D)$
FABOA	124.8	426	192	52	81	$O(N \times T \times D)$
BOA-DQN	132.7	548	176	56	73	$O(N \times T \times D + E)$
FABOA-DQN	136.9	566	152	62	66	$O(N \times T \times D + E)$
FABOA-DDQN	139.5	582	124	69	58	$O(N \times T \times D + E)$

Table 6. Ablation Study Results

Configuration	Optimization Strategy	Reinforcement Learning	Selected Features	Feature Reduction (%)	Predictive Performance (%)	Fitness Value	Runtime (s)
BOA	Standard BOA	–	248	38	89.30	0.284	118.4
FABOA	Fuzzy Adaptive BOA	–	192	52	92.70	0.198	124.8
BOA-DQN	Standard BOA	DQN	176	56	93.90	0.176	132.7
FABOA-DQN	Fuzzy Adaptive BOA	DQN	152	62	95.10	0.158	136.9
FABOA-DDQN	Fuzzy Adaptive BOA	DDQN	124	69	96.80	0.142	139.5

The ablation study, runtime evaluation, convergence analysis, and statistical significance evaluation are reported fully numerically to enhance the verifiability of the experimental analysis. The main variants are original BOA, FABOA with fuzzy parameter adjustment, BOA with integrated DQN-based policy guidance, and complete FABOA-DDQN framework, with clearly defined variants. All variants are tested with the same universe, pre-processing, population size, number of iterations, and settings for the classifiers to measure the contribution of each component. The runtime analysis are reporting the average time, computational complexity and memory usage over multiple independent times, whereas the convergence analysis is showing fitness progression curves and number of iterations needed to obtain the stable performance. Each experiment is repeated 30 times with independent random initializations and the results are shown as mean $\pm$ standard deviation to ensure robustness against random variations. The data is analyzed statistically using appropriate hypothesis testing between FABOA-DDQN with the baseline methods at the level of 95% confidence ($p < 0.05$). Table 6 shows the ablation study results.

Fuzzy adaptive optimisation and Double Deep Q-Network learning complement each other to make the FABOA-DDQN framework perform better than others. Traditional wrapper-based feature selection approaches use preset exploration and exploitation tactics that converge prematurely or trap in local optima for high-dimensional data. The proposed FABOA uses a fuzzy adaptive mechanism to dynamically adjust the exploration-exploitation balance according to the optimisation state, making it more effective at finding discriminative feature subsets while maintaining convergence stability. This adaptive search improves feature quality and reduces redundancy. Unlike RLNMFA, which improves clustering by matrix factorisation and graph regularisation, FABOA-DDQN

optimises feature subsets before learning, giving the classifier more informative and compact feature representations. Deterministic gradient-based optimisation optimises deep matrix decomposition in DBMD-DLM-BFGS. It clusters competitively but lacks adaptive sequential feature subset selection. FABOA-DDQN improves Precision, Recall, F1-score, and Rand Index by 1.95%, 3.44%, 2.69%, and 3.29% across all datasets. The system also outperforms DQN-based feature selection approaches. Standard DQN algorithms overestimate Q-value, causing unstable policy updates and inconsistent feature selection. The model decreases estimate bias and stabilises policy learning by using Double Deep Q-Network (DDQN) learning with distinct online and target networks, improving feature selection and classification. The fuzzy adaptive optimization identifies high-quality candidate feature subsets, while DDQN effectively learns the optimal feature selection policy from these candidates. The combination of these two components produces greater performance gains than either optimization or reinforcement learning alone, demonstrating that the observed improvements arise from the synergy between adaptive fuzzy optimization and bias-reduced reinforcement learning.

To evaluate the robustness and reproducibility of the proposed FABOA-DDQN framework, all experiments were independently repeated 30 times using different random initialization seeds (seed values: 1001–1030) while maintaining identical preprocessing procedures, parameter settings, and stopping criteria for all competing methods. The reported performance values correspond to the mean $\pm$ standard deviation over the 30 independent runs. The proposed FABOA-DDQN achieved a Precision of $98.34 \pm 0.29\%$, Recall of $96.77 \pm 0.35\%$, F1-score of $97.55 \pm 0.31\%$, and Rand Index of $97.10 \pm 0.28\%$ on the Covtype dataset. Statistical comparisons with the strongest baseline (DBMD-DLM-BFGS) were performed using a paired two-tailed t-test at a significance level of $\alpha = 0.05$. The analysis yielded a t-

statistic of 6.84, an exact p-value of 2.7×10^{-6} , and a 95% confidence interval of [0.62, 1.41] for the improvement in F1-score. Since multiple pairwise comparisons were conducted, the Bonferroni correction was applied, resulting in an adjusted significance threshold of $\alpha = 0.0071$ (0.05/7). The observed improvement remained statistically significant after correction (adjusted $p = 1.89 \times 10^{-5} < 0.0071$), confirming that the performance gains achieved by the proposed FABOA-DDQN framework are statistically reliable.

5. Discussion

The experimental results confirm prior findings and show the benefits of merging adaptive optimisation with reinforcement learning. Conventional Nonnegative Matrix Factorisation (NMF) methods learn latent representations from high-dimensional data to improve clustering, but they are sensitive to redundant features and use fixed optimisation strategies that may limit clustering in complex datasets [17]. The Self-Supervised Semi-Supervised Nonnegative Matrix Factorisation (S4NMF) framework uses labelled and unlabelled data to produce more discriminative latent representations, improving clustering accuracy [32]. Adaptive graph regularisation preserves local manifold structures during matrix factorisation in RLNMFA, improving resilience [33]. These approaches increase representation learning but do not optimise feature subset selection, which gives the FABOA-DDQN framework improved feature reduction and predictive performance. Das et al.'s Distributed Bayesian Matrix Decomposition (DBMD) technique is extended by the experiments [18]. The DBMD variations increase clustering by distributed matrix decomposition and optimisation, although they use deterministic optimisation algorithms and lack an adaptive sequential feature subset assessment mechanism. In contrast, the proposed FABOA dynamically changes the exploration–exploitation balance utilising fuzzy adaptive control, improving high-dimensional search space navigation and decreasing premature convergence. Recent investigations have shown that adaptive parameter management improves metaheuristic optimisation [25]. Early reinforcement learning-based feature selection research is supported by the advances. Wang et al. [20] and Alavizadeh et al. [21] showed that Deep Q-Networks can learn feature selection and decision policies, however typical DQN models overestimate Q-value, resulting in unstable policy updates. Recent research has focused on enhanced DQN designs for large-scale adaptive optimisation and decision-making [24, 26, 27]. DQN learning, where distinct online and target networks minimise estimate bias and increase policy stability, is added to these techniques. Thus, the reinforcement learning module optimises feature subsets for more information while retaining convergence. While previous wrapper-based feature selection methods [23, 30] and

FDR²-BD [31] mainly reduce feature dimensionality, the proposed framework optimises adaptive feature subset creation and reinforcement learning-guided assessment. This complementary integration improves feature selection frameworks for scalable big data analytics by compacting feature subsets and retaining predictive performance. The present findings support earlier findings on adaptive optimisation and reinforcement learning and show that integrating them into the FABOA-DDQN framework improves convergence stability, feature reduction efficiency, and predictive performance. The RL-enhanced MGO framework in [37] demonstrates that learned optimization decisions can improve metaheuristic search, while the RL-based evolutionary FS approach in [38] confirms the value of combining learned selection strategies with evolutionary optimization. In comparison, FABOA-DDQN integrates fuzzy adaptive search control and DDQN-based decision learning, providing a coordinated mechanism for balancing exploration, exploitation, and feature-subset quality.

The FABOA-DDQN architecture for wrapper-based feature selection in large data analytics performs well, however it has numerous drawbacks. First, wrapper-based optimisation involves frequent downstream learning model assessments of potential feature subsets, which is more computationally expensive than filter-based feature selection approaches, especially for high-dimensional datasets. Second, the fuzzy adaptation mechanism design and hyperparameters like population size, learning rate, fuzzy adaptation coefficient, and DDQN training parameters affect optimisation performance. Optimisation stability is improved by parameter adjustment, however convergence behaviour and prediction performance may vary. Third, the downstream classifier used to analyse feature subsets determines the outcomes. To guarantee fairness, all competing techniques used a same assessment process, yet various classifiers may perform differently. Despite their popularity, three benchmark datasets were utilised for experimental validation, which may not completely reflect large-scale real-world applications.

6. Conclusion and Future Work

Big data analytics wrapper-based feature selection using FABOA-DDQN was suggested in this research. Fuzzy adaptive control dynamically balances exploration and exploitation during optimisation in the proposed system. An adaptive wrapper goal function optimised predictive performance, feature sparsity, and redundancy reduction. Only the DDQN module was used as a reinforcement learning controller to drive feature selection via policy learning, enhancing optimisation stability and convergence efficiency. On the Covertype, KDD-99, and SUSY benchmark datasets, the proposed framework outperformed baseline

techniques in predictive performance and feature reduction. The ablation investigation showed that fuzzy adaptive optimisation with DDQN policy guidance improved performance, while the convergence analysis showed quicker optimisation with fewer iterations. The computational study also indicated that the proposed framework yields more compact feature subsets with increased optimisation efficiency, despite more computational cost from reinforcement learning than standard metaheuristic techniques. Some restrictions should be noted despite these promising findings. Because the downstream classifier must repeatedly assess possible feature subsets, wrapper-based optimisation increases computing cost. DDQN hyperparameters and fuzzy adaption rules affect optimisation performance. However, the experimental validation was confined to three benchmark datasets under offline learning settings, and the framework's applicability to additional application domains and dynamically developing data streams needs more study. Distributed optimisation and lightweight reinforcement learning systems will reduce computing complexity in future research. To enhance streaming and dynamic large data performance, adaptive hyperparameter optimisation, online feature selection, and incremental reinforcement learning will be examined. Further validation on varied real-world datasets from numerous application areas will examine the framework's resilience and scalability under wider operational settings.

References

- [1] H. Zhang, X. Yue, X. Gao, Reinforcement learning guided auto-select optimization algorithm for feature selection. *Expert Systems with Applications*, 268, (2025) 126320. <https://doi.org/10.1016/j.eswa.2024.126320>
- [2] B. Wei, J. Huang, L. Deng, S. Yang, J. Zheng, Y. Huang, Reinforcement learning-based particle swarm optimization with adaptive scoring mechanism for high-dimensional feature selection. *Swarm and Evolutionary Computation*, 98, (2025) 102104. <https://doi.org/10.1016/j.swevo.2025.102104>
- [3] G. Yang, Z. Zeng, X. Pu, R. Duan, Feature selection optimization algorithm based on evolutionary Q-learning. *Information Sciences*, 719, (2025) 122441. <https://doi.org/10.1016/j.ins.2025.122441>
- [4] Y. Vivek, V. Ravi, P. R. Krishna, Online feature subset selection for mining feature streams in big data via incremental learning and evolutionary computation. *Swarm and Evolutionary Computation*, 94, (2025) 101896. <https://doi.org/10.1016/j.swevo.2025.101896>
- [5] S.K.R. Mallidi, R.R. Ramisetty, Bowerbird courtship-inspired feature selection for efficient high-dimensional data analysis using a novel meta-heuristic. *Discover Computing*, 28, (2025) 6. <https://doi.org/10.1007/s10791-025-09497-2>
- [6] J. Huang, X. Deng, L. Hu, Enhanced grey wolf optimizer with hybrid strategies for efficient feature selection in high-dimensional data. *Information Sciences*, 705, (2025) 121958. <https://doi.org/10.1016/j.ins.2025.121958>
- [7] L. Pan, W.L. Cheng, W.H. Lim, A. Sharma, V. Jatelly, S.S. Tiang, A.H. Alharbi, E.S.M. El-kenawy, A robust wrapper-based feature selection technique based on modified teaching learning based optimization with hierarchical learning scheme. *Engineering Science and Technology, an International Journal*, 61, (2025) 101935. <https://doi.org/10.1016/j.jestch.2024.101935>
- [8] W.L. Cheng, W.H. Lim, K.S. Chong, S.S. Tiang, Y.H. Choo, E.S.M. El-Kenawy, A.H. Alharbi, M.M. Eid, An adaptive and parallel metaheuristic framework for Wrapper-Based feature selection using Arctic Puffin optimization, 85(1), (2025) 2021-2050. <https://doi.org/10.32604/cmc.2025.064243>
- [9] H. Gao, Y. Guo, Z. Zhang, Quantum jellyfish search optimizer applied in high-precision wrapper feature selection. *Pattern Recognition*, 172, (2026) 112620. <https://doi.org/10.1016/j.patcog.2025.112620>
- [10] C. Zhong, H. Chen, D. Xin, T. Xu, Z. Meng, X. Wang, A.R. Yildiz, S. Mirjalili, Opposition and reinforcement learning growth-starfish optimization algorithm for engineering design and feature selection. *Knowledge-Based Systems*, 338, (2026) 115522. <https://doi.org/10.1016/j.knosys.2026.115522>
- [11] H. Quan, Y. Zhang, Q. Li, Y. Liu, An explainable feature selection method through statistical reinforcement learning. *Applied Soft Computing*, 197, (2026) 115184. <https://doi.org/10.1016/j.asoc.2026.115184>
- [12] G. Fu, D. Zhao, Y. Li, L. Liu, A. Qi, Y. Chen, H. Chen, A deep reinforcement learning-based Grey Wolf Optimizer with operator adaptation for hyperspectral image band selection. *Knowledge-Based Systems*, 352(116899), (2026). <https://doi.org/10.1016/j.knosys.2026.116899>
- [13] A.K. Shukla, S. Dwivedi, S. Kumari, S.K. Singh, R. Tirumalasetti, Metaheuristic-driven feature selection for enhanced cancer classification. *Array*, 30, (2026) 100747. <https://doi.org/10.1016/j.array.2026.100747>
- [14] M. Batis, Y. Chen, L. Liu, A. A. Heidari, H. Chen, Prey capture enhanced Harris hawks optimizer for wrapper-based feature selection in high-dimensional medical data. *Computer Methods and Programs in Biomedicine*, 277, (2026) 109237. <https://doi.org/10.1016/j.cmpb.2026.109237>

- [15] Y. Liu, P. Gao, L. Wang, J. Sun, W. Gao. Meta-learning-based multi-view multi-label feature selection with streaming labels. *Knowledge-Based Systems*, 344, (2026) 116084. <https://doi.org/10.1016/j.knosys.2026.116084>
- [16] C. Hou, Z. Wang, Y. Zhang, Y. Todo, J. Tang, Z. Lei, S. Gao, Differential evolution algorithm with cosine similarity-based individual reduction and symmetric uncertainty-based attribute recovery for feature selection. *Applied Soft Computing*, 184, (2025) 113764. <https://doi.org/10.1016/j.asoc.2025.113764>
- [17] J. Xu, Y. Wang, X. Xu, K.K. Cheng, D. Raftery, J. Dong, NMF-based approach for missing values imputation of mass spectrometry metabolomics data. *Molecules*, 26(19), (2021) 5787. <https://doi.org/10.3390/molecules26195787>
- [18] B. Das, T. Murari, C. Das, P. Rajeswari, (2023) Big Data Mining And Clustering Using Distributed Bayesian Matrix Decomposition. In 2023 International Conference on Evolutionary Algorithms and Soft Computing Techniques (EASCT), IEEE, India. <https://doi.org/10.1109/EASCT59475.2023.10392816>
- [19] X. Zhou, W. Liang, I. Kevin, K. Wang, L.T. Yang, Deep correlation mining based on hierarchical hybrid networks for heterogeneous big data recommendations. *IEEE Transactions on Computational Social Systems*, 8(1), (2020) 171-178. <https://doi.org/10.1109/TCSS.2020.2987846>
- [20] Q. Wang, Y. Guo, L. Yu, X. Chen, P. Li, Deep Q-network-based feature selection for multisourced data cleaning. *IEEE Internet of Things Journal*, 8(21), (2020) 16153-16164. <https://doi.org/10.1109/IJOT.2020.3016297>
- [21] H. Alavizadeh, H. Alavizadeh, J. Jang-Jaccard, Deep Q-learning based reinforcement learning approach for network intrusion detection. *Computers*, 11(3), (2022) 41. <https://doi.org/10.3390/computers11030041>
- [22] H. Du, P. Han, Q. Xiang, S. Huang, Monkeyking: Adaptive parameter tuning on big data platforms with deep reinforcement learning. *Big Data*, 8(4), (2020) 270-290. <https://doi.org/10.1089/big.2019.0123>
- [23] S. Md Mujeeb, R. Praveen Sam, K. Madhavi, Adaptive exponential bat algorithm and deep learning for big data classification. *Sādhanā*, 46(1), (2021) 15. <https://doi.org/10.1007/s12046-020-01521-z>
- [24] F. Chen, C. Wu, Design of Big Data Task Scheduling Optimization Algorithm Based on Improved Deep Q-Network. *International Journal of Advanced Computer Science & Applications*, 15(2), (2024) 103. <https://doi.org/10.14569/IJACSA.2024.01502103>
- [25] Z. Benmamoun, K. Khlie, G. Bektemyssova, M. Dehghani, Y. Gherabi, Bobcat Optimization Algorithm: an effective bio-inspired metaheuristic algorithm for solving supply chain optimization problems. *Scientific Reports*, 14(1), (2024) 20099.
- [26] U. Ahmed, J.C. W. Lin, G. Srivastava, Y. Djenouri, (2021) A deep q-learning sanitization approach for privacy preserving data mining. In Adjunct Proceedings of the 2021 International Conference on Distributed Computing and Networking, 43-48. <https://doi.org/10.1145/3427477.3429990>
- [27] Y. Ding, L. Ma, J. Ma, M. Suo, L. Tao, Y. Cheng, and C. Lu, Intelligent fault diagnosis for rotating machinery using deep Q-network based health state classification: A deep reinforcement learning approach. *Advanced Engineering Informatics*, 42, (2019) 100977. <https://doi.org/10.1016/j.aei.2019.100977>
- [28] R. H. Moulton, J. Zgraja, (2019) The Wilderness Area Data Set: Adapting the Covtype data set for unsupervised learning. *arXiv preprint arXiv:1901.11040*. <https://doi.org/10.48550/arXiv.1901.11040>
- [29] S. Gurung, M.K. Ghose, A. Subedi, Deep learning approach on network intrusion detection system using NSL-KDD dataset. *International Journal of Computer Network and Information Security*, 11(3), (2019) 8-14. <https://doi.org/10.5815/ijcnis.2019.03.02>
- [30] S. Kamble, A.J. Siddegowda, V.K. Rajuk. (2023) Wrapper-Based Feature Selection for Big Data Using a Multi-Core Environment. In *Research Advances in Intelligent Computing*, CRC Press. <https://doi.org/10.1201/9781003320340-11>
- [31] M.J. Basgall, M. Naiouf, A. Fernández, FDR2-BD: A fast data reduction recommendation tool for tabular big data classification problems. *Electronics*, 10(15), (2021) 1757. <https://doi.org/10.3390/electronics10151757>
- [32] J. Chavoshinejad, S.A. Seyedi, F.A. Tab, N. Salahan, Self-supervised semi-supervised nonnegative matrix factorization for data clustering. *Pattern Recognition*, 137, (2023) 109282. <https://doi.org/10.1016/j.patcog.2022.109282>
- [33] J. Tang, H. Feng, Robust local-coordinate non-negative matrix factorization with adaptive graph for robust clustering. *Information Sciences*, 610, (2022) 1058-1077. <https://doi.org/10.1016/j.ins.2022.08.023>
- [34] U. Mlakar, I. Fister Jr., I. Fister, Threshold adaptation for improved wrapper-based evolutionary feature selection. *Biomimetics*, 10(10), (2025) 670. <https://doi.org/10.3390/biomimetics10100670>
- [35] Y. Shi, Y. Zheng, X. Bai, A multiple filter-wrapper

feature selection algorithm based on process optimization mechanism for high-dimensional omics data analysis. PLOS ONE, 20, (2025) e0338051.

<https://doi.org/10.1371/journal.pone.0338051>

- [36] C. Hou, Z. Yuan, Y. Zhang, Z. Lei, S. Gao, Temporal difference reinforcement learning-based ant colony optimization with extremized probability construction for feature selection. Results in Engineering, (2026) 109721. <https://doi.org/10.1016/j.rineng.2026.109721>
- [37] H. Song, L. Liu, Y. Chen, Q. Zhang H. Chen, QMGO: A reinforcement learning decision-making enhanced moss growth optimizer for feature selection. International Journal of Computational Intelligence Systems, 19, (2026) 10. <https://doi.org/10.1007/s44196-025-01032-x>
- [38] M. Ramezankhani, A. Boghosian, S.J. Mousavirad, A Novel Reinforcement Learning-Based Feature Selection with Scope Loss and Evolutionary Optimization for Accurate Real Estate Tax Forecasting. International Journal of Computational Intelligence Systems, 19(1), (2026) 67. <https://doi.org/10.1007/s44196-025-01134-6>

Authors Contribution Statement

Both authors contributed equally to this work, and both have read and agreed to the published version of the manuscript.

Funding

The authors declare that no funds, grants or any other support were received during the preparation of this manuscript.

Competing Interests

The authors declare that there are no conflicts of interest regarding the publication of this manuscript.

Data Availability

The data supporting the findings of this study can be obtained from the corresponding author upon reasonable request.

Has this article screened for similarity?

Yes

About the License

© The Author(s) 2026. The text of this article is open access and licensed under a Creative Commons Attribution 4.0 International License.