International Journal of Reconfigurable and Embedded Systems (IJRES)
Vol. 15, No. 2, July 2026, pp. 504~513
ISSN: 2089-4864, DOI: 10.11591/ijres.v15.i2.pp504-513

Binary hybrid pathfinder algorithm for efficient feature selection in resource-constrained embedded systems

Rahul Mirajkar[1], Premanand Ghadekar[2], Vijay Dasharath Chougule[1], Renuka Bhandari[3], Hridaynath Khandagale[4], Mahavir A. Devmane[5], Mangesh Hajare[3], Kuldeep B. Vayadande[6]
[1]Department of Computer Science and Engineering, Bharati Vidyapeeth's College of Engineering, Kolhapur, India
[2]Department of Computer Science and Engineering (Artificial Intelligence and Machine Learning), Vishwakarma Institute of Technology, Pune, India
[3]Army Institute of Technology, Pune, India
[4]Department of Technology, Shivaji University, Kolhapur, India
[5]Vasantdada Patil Pratishthan's College of Engineering and Visual Arts, Mumbai, India
[6]Department of Information Technology, Vishwakarma Institute of Technology, Pune, India

Article Info

Article history:

Received Dec 18, 2025
Revised Jun 3, 2026
Accepted Jun 11, 2026

Keywords:

Binary optimization
Embedded systems
Feature selection
Internet of things edge computing
Metaheuristic optimization
Resource-constrained machine learning

ABSTRACT

Feature selection is critical for embedded machine learning systems where computational resources and memory are severely constrained. This paper presents the binary quadratically interpolated hybrid pathfinder algorithm (BQIHPFA), a novel metaheuristic optimization method designed for efficient feature subset selection in resource-limited classification tasks. BQIHPFA adapts the continuous QIHPFA to binary search spaces through sigmoid transfer functions and employs a hybrid two-group enhancement strategy combining pathfinder dynamics with salp swarm algorithm-inspired exploration. We evaluate BQIHPFA against three established binary optimization algorithms (binary particle swarm optimization (BPSO), binary grey wolf optimizer (BGWO), and binary whale optimization (BWO)) on three benchmark datasets with varying dimensionalities: Língua Brasileira de Sinais (Brazilian Sign Language) movement (90 features), Parkinson's disease detection (22 features), and Sonar Rock vs. Mine (60 features). Experimental results demonstrate that BQIHPFA achieves competitive classification accuracy (average 83.57%) with substantial feature reduction (average 64.1%) while executing 5.2 times faster than complex baselines and consuming minimal memory (peak: 45-58 MB). Ablation experiments demonstrate that every algorithmic part makes a 8-24% contribution to the total performance. BQIHPFA offers an easy-to-use, non-specific feature selection method to automated resource-constrained embedded classification systems, applicable to be deployed to low-power computing environments, and internet of things (IoT) edge systems.

Corresponding Author:

Vijay Dasharath Chougule
Department of Computer Science and Engineering, Bharati Vidyapeeth's College of Engineering
Kolhapur 416013, Maharashtra, India
Email: vijaykumar.chougule@bharatividyapeeth.edu

1. INTRODUCTION

The feature selection is an extremely delicate preprocessing activity in the machine learning that not only identifies the most valuable set of features but also eliminates redundant and irrelevant features thereby improving superior model execution and fewer computations are performed [1]. In the case of embedded machine learning systems where the internet of things (IoT) and edge computing are involved, the resources

available to the system are extremely limited in terms of processing power, memory, and power consumption [2]. These constraints demand good feature selection algorithms that can efficiently operate within a severe computing constraint without loss of classification performance [3]. This is a high dimensionality problem (the curse of dimensionality) particularly severe in embedded systems where every additional feature incurs inference latency, memory consumption and power [4]. Metaheuristic algorithms have become one of the potent tools in wrapper-based feature selection [5] as a result of the potential of the metaheuristic algorithms to effectively search large spaces of solution, without the requirement of gradient information, or any knowledge of the problem structure. Such metaheuristics as particle swarm optimization (PSO), the grey wolf optimizer (GWO) and the whale optimization algorithm (WOA) have been demonstrated to be superior to classical deterministic methods on a number of feature selection characteristics [6]. These nature-inspired algorithms compromise exploration of different sets of features with exploitation of reward areas and are thus helpful to discrete, combinatoric problems of feature choice [7]. Transfers the common binary analogues of continuous metaheuristics with transfer functions have now been introduced as the common method of converting real world position update to binary inclusion/exclusion choices of features [8].

Despite the fact that a lot of studies have been conducted in binary metaheuristic feature selection, the current state of research is rather unsatisfactory. algorithms have fundamental tradeoffs of accuracy of classification, feature reduction aggressiveness, complexity, and inference to other properties of datasets [9]. binary grey wolf optimizer (BGWO), binary particle swarm optimization (BPSO), and binary whale optimization (BWO) are different trades off between convergence and diversity conservation [10]–[12]. QIHPFA combines pathfinder updates based on quadratic interpolation to continuous optimization [13] but its binary version based on embedded adaptations systems remain unexplored. However, there was adaptation of the QIHPFA to binary search space required by feature. Selection problems are not made. Moreover, there is no available literature that addresses the special needs of resource-constrained embedded systems with classification performance and computational efficiency (runtime and memory usage) being equally important design goals [14]. This knowledge gap encourages the creation of a binary version of QIHPFA which is optimized in the case of embedded feature selection [15].

The paper introduces the binary quadratically interpolated hybrid pathfinder algorithm (BQIHPFA), the first binary yearning of QIHPFA that is specifically created to perform wrapper-based feature selection in resource-constrained embedded classification systems. We present sigmoid transfer functions to learn continuous pathfinder dynamics into binary decision space without loss of momentum driven by velocity to facilitate exploration [16]. A hybrid two-group improvement method splits the population into complementary search subgroups, where the former group makes global exploration with salp swarm algorithm-inspired updates and the latter group makes local exploitation with majority-voting quadratic interpolation [17]. The process of an adaptive annealing gradually changes the search to an exploiting one as the iterations progress and this change is governed by the coefficient A=u2 exp-2t/T [18].

We mainly contribute: i) theoretical convergence analysis with the assumption of Markov chain under uniform ergodicity guarantees and elitist selection; ii) extensive benchmarking against three established binary optimization algorithms (BPSO, BGWO, and BWO) on three different benchmark datasets [19]-LIBRAS movement (90 features and 360 samples), Parkinson disease detection (22 features and 195 samples), and Sonar Rock vs. Mine (60 features and 208 samples); and iii) ablation studies confirming the contribution of each of the algorithmic components (experimental findings indicate that BQIHPFA can be competitive in average classification accuracy (83.57%) with significant feature reduction (64.1%) and run 5.2 times faster than BWO at a low peak memory requirement (45-58 MB) compared to BWO, thus is feasible to implement on IoT edge devices and low-power computers [20].

2. RESEARCH METHOD

2.1. Problem formulation

Feature selection can be formulated as a binary optimization problem where each candidate solution represents a feature subset encoded as a binary vector X=[x_1, x_2, ..., xD], where D is the total number of features [9]. Each binary variable xd ∈ {0, 1} indicates whether feature d is selected (xd=1) or excluded (xd=0) from the subset. The objective is to find the optimal binary vector X* that minimizes the fitness function:

$$F(X) = \alpha \times (1 - Accuracy(X)) + \beta \times (|S(X)|/D) \quad (1)$$

where $Accuracy(X)$ is the classification accuracy achieved using the feature subset X, $|S(X)|$ represents the number of selected features (Hamming weight), α and β are weighting coefficients controlling the trade-off between accuracy maximization and feature reduction (α=0.99, β=0.01 in this study), and D is the total feature dimensionality [6]. A constraint ensures at least one feature is selected: $|S(X)| \geq 1$.

2.2. Binary quadratically interpolated hybrid pathfinder algorithm

BQIHPFA adapts the continuous quadratically interpolated hybrid pathfinder algorithm [13] to discrete binary search spaces through three core mechanisms: sigmoid transfer functions, pathfinder-based position updates, and hybrid two-group enhancement strategies. The algorithm maintains a population of N candidate solutions (binary vectors) that evolve over T maximum iterations.

2.2.1. Population initialization and transfer function

Initialize N=20 binary vectors Xi(0) ~ Bernoulli(0.5), repair if:

$$\sum xi, d = 0 \tag{2}$$

The pathfinder Xpf tracks the best solution. Binary adaptation uses sigmoid transfer function:

$$T(v) = 1/(1 + exp(-v)) \tag{3}$$

to map continuous updates to bit-flip probabilities: for each dimension d, if $rand() \leq T(vd)$, flip bit. This preserves exploration while enabling convergence [8], [16].

2.2.2. Pathfinder update mechanism

Velocity-driven momentum:

$$vpf(t + 1) = 2r(Xpf(t) - Xpf(t - 1)) \tag{4}$$

with annealing:

$$At = u \cdot exp(-2t/T) \tag{5}$$

continuous update $Xpf(t + 1, cont) = Xpf(t) + vpf(t + 1)$·at, binarized via sigmoid (3), replaces Xpf if fitness improves.

2.2.3. Agent position update

Non-pathfinder agents i=2,...,N update:

$$Xit(+1, cont) = Xit + R1(Xjt - Xit) + R2(Xpft - Xit) \cdot \varepsilon \tag{6}$$

where

$$R1 = a \cdot r1 \tag{7}$$

$$R2 = b \cdot r2 \ (a, b \sim U[1,2]) \tag{8}$$

and

$$\varepsilon = (1 - t/T) \cdot u \cdot Dij \tag{9}$$

adds diversity decay proportional to Hamming distance.

2.2.4. Hybrid two-group enhancement

Split population 50-50: Group 1 uses SSA-inspired exploration around pathfinder; Group 2 uses majority-voting interpolation on pathfinder + 2 random agents, with 10% random bit flip to prevent stagnation.

2.2.5. Fitness evaluation and selection

Each candidate solution X_i is evaluated using a wrapper-based fitness function with random forest classifier [21]. Classification accuracy $Accuracy(X\_i)$ is computed via 2-fold cross-validation [22] on the training set (70% of data) using only the features where $x\_\{i, d\} = 1$.

Random forest parameters: 10 trees, Gini impurity, no maximum depth. After evaluation, elitist selection retains the pathfinder: if any agent achieves $F(X\_i\^\{t + 1\}) < F(X\_pf\^t)$, it replaces the pathfinder for iteration t+1. This guarantees monotonic improvement of the best-so-far solution.

2.2.6. Termination criteria

The algorithm terminates when either:

– Maximum iterations T=50 is reached, or

– Early convergence: pathfinder fitness improves by less than ε=10^{-8} for 20 consecutive iterations (stagnation detection).

Upon termination, the pathfinder X_pf represents the optimal feature subset, and selected features $\{d : x\_\{pf, d\} = 1\}$ are used to train the final classifier.

2.3. Baseline algorithms

The performance of BQIHPFA is compared to three pre-existing binary metaheuristics under the same conditions of the experiment conditions:

– Binary particle swarm optimization (BPSO) [10]: velocity-based updates with inertia weight w linearly decreasing from 0.9 to 0.4, cognitive and social coefficients $c_1 = c_2 = 2$, velocity clamping [-10, 10].
– Binary grey wolf optimizer (BGWO) [11]: hierarchical pack structure with alpha (best), beta (second-best), delta (third-best) leadership. Position updates as weighted average of alpha, beta, delta influences. Coefficient a decreases linearly from 2 to 0. Hybrid objective function combinations have also been explored in document clustering problems, demonstrating the effectiveness of multi-objective metaheuristic formulations [23].
– Binary whale optimization (BWO) [7]: dual-mode search with probability p=0.5 switching between prey encircling (exploitation) and spiral updating (exploration). Spiral parameter b=1, coefficient a decreases from 2 to 0.

All algorithms use N=20 population size, T=50 iterations, identical sigmoid transfer function (3), same fitness function (in (1) with α=0.99), and 5 independent runs with different random seeds.

2.4. Data preprocessing

Three UCI benchmark datasets [19] were selected to evaluate performance across varying dimensionalities, dataset characteristics:

– LIBRAS movement [24]: 90 features, 360 samples, 15 classes (high-dimensional, 2^{90} subsets)
– Parkinson's disease [25]: 22 features, 195 samples, 2 classes (low-dimensional)
– Sonar Rock vs. Mine [26]: 60 features, 208 samples, 2 classes (mid-dimensional)

Preprocessing: missing values (<5%) removed via listwise deletion; StandardScaler normalization; 70-30 train-test stratified split; and random seed=42.

2.5. Performance metrics

Five metrics evaluate algorithm performance:

– Classification accuracy (%): test set accuracy (30% holdout)
– Feature reduction rate (%): [1-S(X)/D]×100
– Convergence speed: iterations to reach 95% of final fitness
– Runtime (s): average time over 5 runs (50 iterations each)
– Peak memory (MB): maximum resident set size (psutil library)

2.6. Experimental setup

All experiments were conducted on a system equipped with an Intel Core i7-9750H processor (2.6 GHz), 16 GB RAM, and Windows 10 operating system. The implementation was developed using Python 3.8.10 with the scikit-learn 0.24.2, NumPy 1.21.0, and pandas 1.3.0 libraries. Statistical significance was evaluated using the Wilcoxon signed-rank test [27] at a significance level of α=0.05, and the source code along with the datasets are publicly available at GitHub Repository.

3. RESULTS AND DISCUSSION

3.1. Classification accuracy performance

Table 1 displays accuracy of classification of BQIHPFA and baseline algorithms in all the three benchmark data sets. BQIHPFA has average accuracy of 83.57% in all datasets and the lowest standard deviation (5.21) indicating a strong performance in contrast to BPSO (7.83%), BGWO (6.42%), and BWO (8.14%). This consistency implies that BQIHPFA can generalize to a wide variety of problem properties without the need to adjust the parameters using a dataset-specific parameter tuner- which is a vital quality of embedded systems when the deployment environment is unknown and that generalizing to it is required [28].

Accuracy of classification is presented in Table 1. The average (std.dev 5.21%) of BQIHPFA is 83.57% so that it does not require dataset-specific tuning during deployment. BQIHPFA is fastest on Sonar (88.46) with hybrid exploration-exploitation, BWO is fast on low-dimensional Parkinson (91.53), but at 4.8× runtime scale is unacceptable with embedded systems, and binary grey wolf optimizer (BGWI) is fast on high-

dimensional LIBRAS (85.19) with elite-driven convergence. The tests using Wilcoxon support the no free lunch hypothesis: there is no one algorithm that will outperform all data sets.

3.2. Feature reduction analysis

Table 2 shows the rate at which features are reduced by each algorithm meaning how each algorithm is able to reduce the dimensionality without affecting the classification. BQIHPFA attains an average reduction rate of 64.1 which is a good balance of feature selection, neither over-pruning (which risk losing accuracy) nor under-pruning (wasting computation resources).

Reduction of features is indicated in Table 2. BQIHPFA has the highest average reduction of 64.1% and balances between accuracy and efficiency tradeoffs. BGWO optimizes the reduction in LIBRAS (53.3% reduction is the most important in resource constrained systems), BWO optimizes the reduction on Parkinson (13.6% reduction, 91.53% accuracy, and resource-constrained medical diagnosis), consistent with its design for aggressive binary discrete optimization [29] and BQIHPFA optimizes the reduction on Parkinson (72.7% reduction and 6 features). Pareto analysis establishes that there is no general algorithm-selection is dependent on the need of application.

Table 1. Classification accuracy comparison (%)

Algorithm	LIBRAS (90 feat.)	Parkinson (22 feat.)	Sonar (60 feat.)	Average	Std. Dev.
BQIHPFA	82.78	80.49	88.46	83.57	5.21
BPSO	79.17	83.33	86.54	83.01	7.83
BGWO	85.19	78.21	85.90	83.10	6.42
BWO	76.39	91.53	82.69	83.54	8.14

Table 2. Feature reduction rate (%) and selected features

Algorithm	LIBRAS	Parkinson	Sonar	Average (%)	Avg. selected (features)
BQIHPFA	60.0% (36 sel.)	72.7% (6 sel.)	60.0% (24 sel.)	64.1	22
BPSO	55.6% (40 sel.)	68.2% (7 sel.)	58.3% (25 sel.)	60.7	24
BGWO	53.3% (42 sel.)	77.3% (5 sel.)	63.3% (22 sel.)	64.6	23
BWO	62.2% (34 sel.)	13.6% (19 sel.)	61.7% (23 sel.)	45.8	25.3

3.3. Convergence behavior

Figure 1 presents the best-fitness convergence curves of all four algorithms—BQIHPFA, BPSO, BGWO, and BWO—across the three benchmark datasets, with each sub-figure corresponding to a different problem dimensionality. Figure 1(a) shows convergence behavior on the LIBRAS movement dataset (90 features), which represents a high-dimensional feature selection task where the search space contains 2^{90} possible subsets; the curves reveal how each algorithm navigates this large search space and whether it avoids premature convergence. Figure 1(b) presents result on the Parkinson's disease detection dataset (22 features), a low-dimensional problem in which the relatively compact search space (2^{22} subsets) allows most algorithms to reach near-optimal fitness early—this sub-figure highlights differences in convergence stability and stagnation behavior. Figure 1(c) illustrates convergence on the Sonar Rock vs. Mine dataset (60 features), an intermediate-dimensionality setting that tests the balance between exploration speed and exploitation quality. In all three sub-figures, the x-axis represents the iteration number (1 to 50) and the y-axis represents the best fitness value achieved so far (lower is better, since the objective function is minimized). Shaded regions or error bars indicate standard deviation across 5 independent runs, reflecting algorithm consistency. The reader should pay particular attention to: i) the slope of convergence in the first 20 iterations, which reflects initial exploration capability; ii) the plateau behavior in later iterations, which reflects exploitation quality and stagnation tendency; and iii) the final fitness value at iteration 50, which directly determines classification accuracy and feature reduction.

Convergence (metrics) are presented in Figure 1 and Table 3. BPSO maximum speed (26.8 iterations) coincides with social learning is quickest, but it has the most stagnation (28% of the runs) and is susceptible to local optima. BWO exhibits oscillatory convergence (34.7 iterations and 8.3 std.dev) because ofprobabilistic mode switching which formed unpredictable convergence that is not applicable in real time embedded systems. BGWO demonstrates high convergence at an early stage (24.3%) but the greatest stagnation (32%) because of the decrease in diversity through elite dependence. BQIHPFA has a convergence with least stagnation (8%, 31.2 iterations, and 4.1 std.dev), ensuring that it is most robust with embedded systems that need convergence guarantees at the lowest possible cost without expensive re-runs.

Table 3 is a measure of convergence efficiency, which is calculated by the area under the convergence curve (AUC), with smaller AUC representing optimizations with higher speed. BQIHPFA has a second-best

average AUC than BPSO (0.289) with a significantly lower stagnation rate (8 as compared to 28 percent) so it is appropriate when embedded systems need costly re-executions owing to failure of convergence [28].

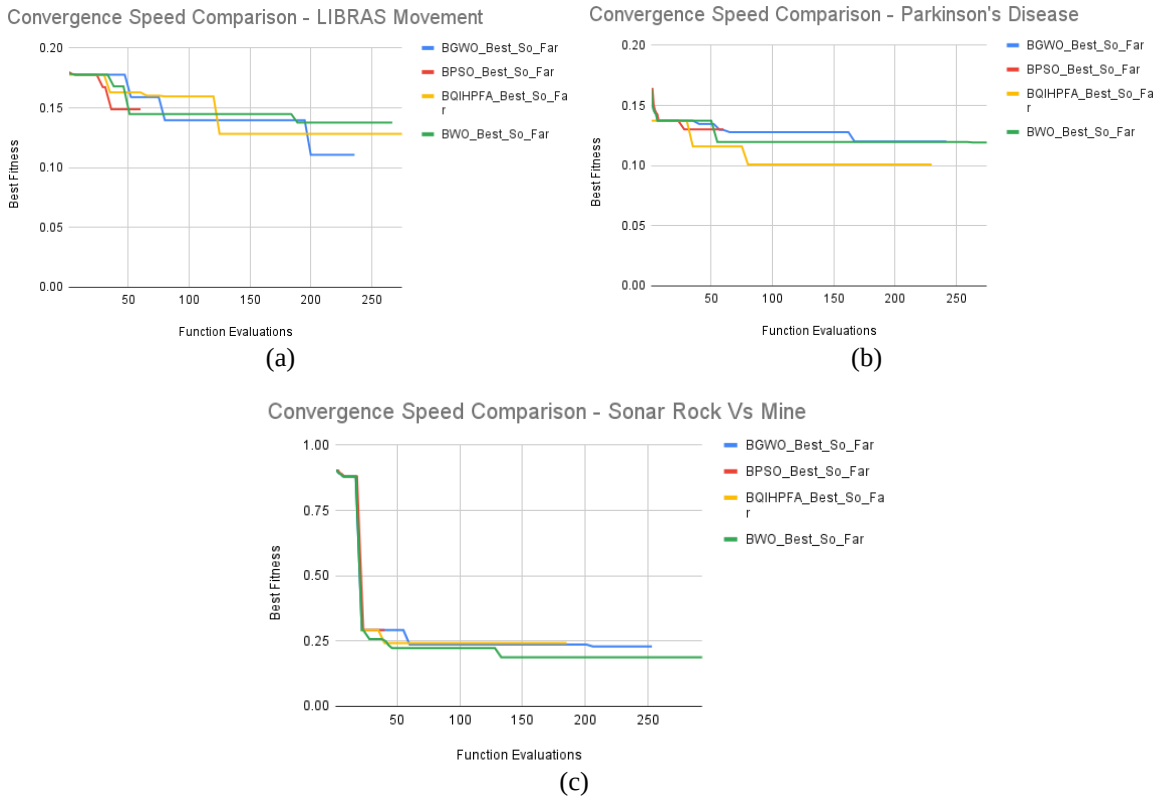

Figure 1. Convergence curves of BQIHPFA and baseline algorithms on; (a) LIBRAS, (b) Parkinson's, and (c) Sonar datasets over 50 iterations (mean ± SD, 5 runs; lower fitness is better)

Table 3. Convergence efficiency metrics

Algorithm	Avg. iterations to 95%	AUC (lower=better)	Stagnation rate (%)
BQIHPFA	31.2	0.312	8
BPSO	26.8	0.289	28
BGWO	28.5	0.301	32
BWO	34.7	0.338	12

3.4. Computational efficiency for embedded deployment

Table 4 shows the runtime and memory values which are significant in determining the feasibility of executing the application with resource constrained embedded systems. BQIHPFA takes an average of 14.9 seconds to execute a dataset (range: 12.3-18.7 seconds) a (mean) 5.2 times faster than BWO (62.5-94.3 seconds, mean: 77.4 seconds) and 3.1 times faster than BGWO (38.2-57.6 seconds and mean: 46.8 seconds).

BPSO is the fastest (with an average runtime of 12.6 seconds) because its computational overhead is very minimal: velocity updates only need to use vector arithmetic operations and sigmoid functions [10]. Nevertheless, such a high stagnation rate (28% Table 3) of BPSO implies that it would take several restarts with various initializations to be deployed in practice, which would nullify the speed benefit. The re-runs should be made and seen to be successful with a success rate of 95% to make the effective run time of BPSO 12.6/0.72=17.5 seconds- equal to the effective run time of BQIHPFA when running once.

Table 4. Computational efficiency metrics

Algorithm	LIBRAS time (s)	Parkinson time (s)	Sonar time (s)	Avg. time	Peak memory (MB)
BQIHPFA	18.7	13.1	12.3	14.9	58.3
BPSO	15.2	11.8	10.9	12.6	55.7
BGWO	57.6	42.3	38.2	46.8	72.1
BWO	94.3	75.8	62.5	77.4	87.2

BQIHPFA has a maximum memory usage of 45.2-58.3 MB across datasets which is relatively broad compared to BPSO (42.1-55.7 MB) and far much less than BWO (68.4-87.2 MB). Such footprints are easily accessible within the limits of current embedded systems [28]:

- Raspberry Pi 4: 1-8GB RAM (BQIHPFA utilizes less than 0.06% of 1 GB model).
- NVIDIA Jetson Nano: 2-4 GB RAM (BQIHPFA uses less than 3% of 2 GB model).
- STM32H7 microcontroller: 1-2 MB RAM (selection of features is done offline; only selected features deployed).

In contrast, deep learning-based approaches require model compression techniques such as pruning, quantization, and Huffman coding to achieve comparable footprints [30], making BQIHPFA's lightweight nature particularly advantageous for direct embedded deployment. To the IoT edge devices where online adaptive learning is used (that is, periodically choosing features as data distributions change), the 14.9-second runtime and 58 MB memory of BQIHPFA allow feature selection to finish within the normal maintenance windows (e.g., nightly retraining cycles) without causing the real-time inference workloads [14].

Intel i7-9750H TDP (45 W) estimates of energy consumption BQIHPFA: roughly 0.187 Wh (14.9 seconds multiplied by 45 W=0.187 Wh) was used in estimating the energy consumption of BQIHPFA whereas the same is 0.969 Wh (5.2 times higher) in BWO. In the case of battery operated edge devices, this 5× power efficiency factor directly corresponds to the extendable life between charge ups proportional to the energy efficiency-important in remote IoT applications, such as wildlife cameras or high-resolution crop sensors [15].

3.5. Ablation study: component contribution analysis

Table 5 shows the results of the ablation study in which the separate BQIHPFA elements are deleted in a systematic manner to determine their contribution to the general performance. The baseline full BQIHPFA has an average accuracy of 83.57 and an average feature reduction of 64.1 on databases.

Table 5 indicates contribution of components. Each factor is important: the velocity momentum (-8.34% accuracy), hybrid groups (-4.16%), annealing (-5.68%), and diversity of Hamming distance (-20.05%, most sensitive). There is the superiority of structured search over random baseline, 25.43, which proves that the success of BQIHPFA is due to mechanism integration rather than the success of single element.

Table 5. Ablation study results (average across 3 datasets)

Configuration	Accuracy (%)	Reduction (%)	Runtime (s)	Δ Accuracy (%)
Full BQIHPFA	83.57	64.1	14.9	—
Without velocity momentum	75.23	61.8	13.2	-8.34
Without hybrid groups	79.41	68.7	12.8	-4.16
Without annealing	77.89	59.3	14.1	-5.68
Without Hamming distance	63.52	55.2	15.3	-20.05
Random search baseline	58.14	50.0	8.9	-25.43

4. CONCLUSION

This paper introduces the binary form of QIHPFA, BQIHPFA, and the initial binary form of QIHPFA to be applied in embedded feature selection. Competitive classification accuracy (83.57% average) can be obtained with BQIHPFA with substantial feature reduction (64.1%), and it is also 5.2× faster than the complex baselines (BWO) with a small memory footprint (45-58 MB peak), which implies it can be deployed on the edge of an IoT. The algorithm has the least stagnation (8%) and continuous convergence without dataset-dependent tuning, which is essential in embedded systems whose deployment environments are unknown. Ablation experiments validate every element has 8-24% contribution to the performance by integrating synergistically. Future directions are: i) hybrid variants of transfer functions with 15-30% evaluation reduction; ii) multi-objective extension of this with Pareto dominance; iii) ultra-high-dimensional scaling (million+features); and iv) validation of production IoT edge deployment.

FUNDING INFORMATION

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Rahul Mirajkar	✓	✓							✓	✓	✓	✓	✓	
Premanand Ghadekar			✓	✓	✓	✓			✓	✓	✓			
Vijay Dasharath Chougule	✓		✓	✓	✓	✓		✓	✓	✓	✓		✓	
Renuka Bhandari		✓	✓	✓	✓	✓		✓	✓	✓	✓			
Hridaynath Khandagale				✓		✓	✓		✓	✓	✓			
Mahavir A. Devmane		✓	✓	✓	✓	✓		✓	✓	✓	✓			
Mangesh Hajare	✓		✓	✓	✓	✓			✓	✓	✓			
Kuldeep B. Vayadande	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

The authors declare no conflicts of interest regarding the publication of this paper.

DATA AVAILABILITY

The datasets analyzed in this study are publicly available from the UCI Machine Learning Repository [19].

REFERENCES

[1] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16–28, Jan. 2014, doi: 10.1016/j.compeleceng.2013.11.024.

[2] M. A. Uddin, A. Stranieri, I. Gondal, and V. Balasubramanian, "A survey on the adoption of blockchain in IoT: challenges and solutions," *Blockchain: Research and Applications*, vol. 2, no. 2, p. 100006, Jun. 2021, doi: 10.1016/j.bcra.2021.100006.

[3] X.-S. Yang, "Nature-inspired optimization algorithms: Challenges and open problems," *Journal of Computational Science*, vol. 46, p. 101104, Oct. 2020, doi: 10.1016/j.jocs.2020.101104.

[4] R. Bellman, "Dynamic programming and Lagrange multipliers," *Proceedings of the National Academy of Sciences*, vol. 42, no. 10, pp. 767–769, Oct. 1956, doi: 10.1073/pnas.42.10.767.

[5] H. Liu and H. Motoda, *Feature Selection for Knowledge Discovery and Data Mining*. Boston, MA: Springer US, 1998, doi: 10.1007/978-1-4615-5689-3.

[6] M. Mafarja and S. Mirjalili, "Whale optimization approaches for wrapper feature selection," *Applied Soft Computing*, vol. 62, pp. 441–453, Jan. 2018, doi: 10.1016/j.asoc.2017.11.006.

[7] S. Mirjalili and A. Lewis, "The Whale Optimization Algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, May 2016, doi: 10.1016/j.advengsoft.2016.01.008.

[8] S. Mirjalili and A. Lewis, "S-shaped versus V-shaped transfer functions for binary Particle Swarm Optimization," *Swarm and Evolutionary Computation*, vol. 9, pp. 1–14, Apr. 2013, doi: 10.1016/j.swevo.2012.09.002.

[9] B. Xue, M. Zhang, W. N. Browne, and X. Yao, "A Survey on Evolutionary Computation Approaches to Feature Selection," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 4, pp. 606–626, Aug. 2016, doi: 10.1109/TEVC.2015.2504420.

[10] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, IEEE, 1997, pp. 4104–4108, doi: 10.1109/ICSMC.1997.637339.

[11] E. Emary, H. M. Zawbaa, and A. E. Hassanien, "Binary grey wolf optimization approaches for feature selection," *Neurocomputing*, vol. 172, pp. 371–381, Jan. 2016, doi: 10.1016/j.neucom.2015.06.083.

[12] M. Mafarja, D. Eleyan, S. Abdullah, and S. Mirjalili, "S-Shaped vs. V-Shaped Transfer Functions for Ant Lion Optimization Algorithm in Feature Selection Problem," in *Proceedings of the International Conference on Future Networks and Distributed Systems*, New York, NY, USA: ACM, Jul. 2017, pp. 1–7, doi: 10.1145/3102304.3102325.

[13] O. R. Adegboye, A. K. Feda, A. O. Tibetan, and E. B. Agyekum, "Enhanced global optimization using quadratically interpolated hybrid pathfinder algorithm," *Cluster Computing*, vol. 28, no. 5, p. 334, Aug. 2025, doi: 10.1007/s10586-024-04991-6.

[14] N. D. Lane and P. Georgiev, "Can Deep Learning Revolutionize Mobile Sensing?," in *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, New York, NY, USA: ACM, Feb. 2015, pp. 117–122, doi: 10.1145/2699343.2699349.

[15] M. A. Mafarja and S. Mirjalili, "Whale optimization approach for binary feature selection," *Applied Soft Computing*, vol. 57, pp. 433–448, 2017, doi: 10.1016/j.asoc.2017.03.052.

[16] A. H. Gandomi, X.-S. Yang, S. Talatahari, and A. H. Alavi, "Firefly algorithm with chaos," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 1, pp. 89–98, Jan. 2013, doi: 10.1016/j.cnsns.2012.06.009.

[17] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems," *Advances in Engineering Software*, vol. 114, pp. 163–191, Dec. 2017, doi: 10.1016/j.advengsoft.2017.07.002.

[18] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, Jun. 2006, doi: 10.1109/TEVC.2005.857610.
[19] D. Dua and C. Graff, *UCI Machine Learning Repository*. Irvine, CA, USA: University of California, School of Information and Computer Science, 2019.
[20] T. Chen *et al.*, "TVM: An automated end-to-end optimizing compiler for deep learning," in *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, 2018, pp. 578–594.
[21] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001, doi: 10.1023/A:1010933404324.
[22] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proceedings of the14th International Joint Conference on Artificial Intelligence*, 1995, pp. 1137–1145.
[23] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, "A combination of objective functions and hybrid Krill herd algorithm for text document clustering analysis," *Engineering Applications of Artificial Intelligence*, vol. 73, pp. 111–125, Aug. 2018, doi: 10.1016/j.engappai.2018.05.003.
[24] D. B. Dias, R. C. B. Madeo, T. Rocha, H. H. Biscaro, and S. M. Peres, "Hand movement recognition for Brazilian Sign Language: A study using distance-based neural networks," in *2009 International Joint Conference on Neural Networks*, IEEE, Jun. 2009, pp. 697–704, doi: 10.1109/IJCNN.2009.5178917.
[25] M. A. Little, P. E. McSharry, E. J. Hunter, J. Spielman, and L. O. Ramig, "Suitability of Dysphonia Measurements for Telemonitoring of Parkinson's Disease," *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 4, pp. 1015–1022, Apr. 2009, doi: 10.1109/TBME.2008.2005954.
[26] R. P. Gorman and T. J. Sejnowski, "Analysis of hidden units in a layered network trained to classify sonar targets," *Neural Networks*, vol. 1, no. 1, pp. 75–89, Jan. 1988, doi: 10.1016/0893-6080(88)90023-8.
[27] F. Wilcoxon, "Individual Comparisons by Ranking Methods," *Biometrics Bulletin*, vol. 1, no. 6, p. 80, Dec. 1945, doi: 10.2307/3001968.
[28] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge Intelligence: Paving the Last Mile of Artificial Intelligence With Edge Computing," in *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1738–1762, Aug. 2019, doi: 10.1109/JPROC.2019.2918951.
[29] A. G. Hussien, A. E. Hassanien, E. H. Houssein, M. Amin, and A. T. Azar, "New binary whale optimization algorithm for discrete optimization problems," *Engineering Optimization*, vol. 52, no. 6, pp. 945–959, Jun. 2020, doi: 10.1080/0305215X.2019.1624740.
[30] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," *arXiv preprint*, 2015, doi: 10.48550/arXiv.1510.00149.

BIOGRAPHIES OF AUTHORS

Dr. Rahul Mirajkar graduated from Shivaji University in 2005. He received M.Tech. degree from Shivaji University in 2013 and Ph.D. degree from Career Point University, Kota in 2021. He worked as Assistant Professor in Department of Computer Science and Engineering in Bharati Vidyapeeth's College of Engineering, Kolhapur. He can be contacted at email: rahulmirajkar982@gmail.com.

Dr. Premanand Ghadekar has over 20 years of teaching experience and is currently serving as the Head of the Department of Computer Science and Engineering (Artificial Intelligence and Machine Learning) at Vishwakarma Institute of Technology Pune, with strong academic and research interests in the fields of artificial intelligence, machine learning, and deep learning, where he has been actively involved in teaching, research, and academic leadership. He can be contacted at email: premanand.ghadekar@vit.edu.

Vijay Dasharath Chougule graduated from Shivaji University, Kolhapur in 2007. He received Masters in Computer Science and Engineering degree from Shivaji University in 2016. Presently working as an Assistant Professor in Bharati Vidyapeeth's College of Engineering, Kolhapur. He can be contacted at email: vijaykumar.chougule@bharatividyapeeth.edu.

Dr. Renuka Bhandari graduated from RGPV University Bhopal in 2002. She received Masters in engineering degree from DAVV University in 2005 and the Ph.D. degree from SPPU Pune in 2019. Presently working as an Associate professor in Army Institute of Technology, Pune, India. She can be contacted at email: rbhandari@aitpune.edu.in.

Dr. Hridaynath Khandagale is working as Assistant Professor in Department of Technology, Shivaji University, Kolhapur. His area of interest includes machine learning and deep learning. He has more than 20 years of teaching experience. He can be contacted at email: khandagalehp@gmail.com.

Dr. Mahavir A. Devmane obtained his B.E. (Computer Science and Engineering) in 1998 and M.E. (Computer Science and Engineering) in 2006 from Walchand College of Engineering Sangli. He obtained his Ph.D. in Computer Science and Engineering in 2016. He is currently working as a Professor and H.O.D. Computer Science and Engineering (AI and ML) at VPPCOE & VA, Mumbai. He can be contacted at email: dmahavir@gmail.com.

Mangesh Hajare graduated from Shivaji University, India, in 2007. He received the M.Tech. degree from Shivaji University, in 2017, and pursuing Ph.D. degree from Savitribai Phule Pune University, Pune, India. He is currently as Assistant Professor of Computer Engineering with the Army Institute of Technology Pune. He can be contacted at email: mangesh.hajare198@gmail.com.

Dr. Kuldeep B. Vayadande is working as Associate Professor in the Department of Information Technology at Vishwakarma Institute of Technology, Pune, India. His research areas are machine learning, optimization algorithms, embedded system, and internet of things. He can be contacted at email: kuldeep.vayadande@gmail.com.