Jurnal Teknologi Informasi dan Komunikasi

Volume 16 Issue 2 Year 2025 | Page 1-11 | e-ISSN: 2477-3255 | ISSN: 2086-4884
Received: 13-06-2025 | Revised: 14-07-2025 | Accepted: 16-09-2025

Sensitivity Analysis of Parameter Control in Leukemia Classification Using Variable-Length Particle Swarm Optimization

Siti Ramadhani[1,2], Lestari Handayani[2], Muhammad Fikri[2], Theam Foo Ng[3], Sumayyah Dzulkifly[1], Roziana Ariffin[4,5], *Shir Li Wang[1,6]

[1] Department of Software Engineering and Meta-Technology, Universiti Pendidikan Sultan Idris, Tanjong Malim, Malaysia
[2] Department of Informatics Engineering, Universitas Islam Negeri Sultan Syarif Kasim Riau, Pekanbaru - Indonesia
[3] Centre for Global Sustainability Studies, Universiti Sains Malaysia, Penang - Malaysia
[4] Premier Integrated Lab, Pantai Hospital Ampang, Kuala Lumpur - Malaysia.
[5] Sunway Medical Centre, Subang Jaya, Selangor - Malaysia.
[6] Data Intelligence and Knowledge Management Special Interest Group, Universiti Pendidikan Sultan Idris, Tanjong Malim, Malaysia
*Corespondence: shirli_wang@meta.upsi.edu.my

Abstract: *Machine learning has the potential to support hematologists in classifying leukemia by identifying abnormal chromosomes and specific gene markers. One effective technique for feature selection is Variable-Length Particle Swarm Optimization (VLPSO), where its performance depends heavily on parameter control, specifically the inertia weight (w) and acceleration factors (c), which regulate the search process. In previous VLPSO, static types of parameter control were applied to the c Factor, and the w Factor used time-varying types. Although its results showed good performance in VLPSO, there was no separation in the treatment of training data and test data, leaving a gap in understanding their impacts for real-world applications. This study examines the impact of various parameter control strategies (static, time-varying, and adaptive) on the performance of VLPSO, comparing it with two adaptive parameter control approaches, Adaptive1 and Adaptive2, within the VLPSO framework. Each approach is designed to dynamically adjust the control parameters w and c in distinct ways. The 10-fold cross-validation results show that VLPSO with an Adaptive one-parameter setting achieves better generalization, characterized by low train-test differences, especially in Decision Tree and Naïve Bayes classifiers, albeit with higher variability. Adaptive2 parameter setting of VLPSO offers more consistent results with narrower variability across different settings. Static methods are the least reliable, while time-varying controls show moderate but unstable performance. Adaptive parameter tuning is recommended to improve VLPSO's stability, flexibility, and classification accuracy in biomedical applications. The results provide recommendations for parameter settings using an adaptive approach that has been proven to enhance the performance of VLPSO.*

Keywords: *Leukemia Classification, Variable-Length Particle Swarm Optimization (VLPSO), Adaptive Parameter Control, Generalization Consistency, Parameter Sensitivity Analysis*

1. Introduction

Early detection of leukemia, a type of blood cancer caused by hematological disorders, is critical for improving treatment outcomes and survival rates[1]. Conventional diagnostic methods, such as morphological assessments and MRI scans, are often invasive, time-consuming, and reliant on expert interpretation, which presents challenges for timely and accurate diagnosis [2], [3]. Recent advancements in machine learning (ML), particularly in feature selection using

Variable Length Particle Swarm Optimization (VLPSO), an enhancement of Particle Swarm Optimization (PSO), have shown promising results in supporting leukemia classification[5], with reported accuracies exceeding 80%–95% depending on the classifier[6],[7]. VLPSO allows each particle to adjust its feature subset length dynamically during the optimization process. This flexibility makes VLPSO especially effective for high-dimensional problems, such as gene selection in leukemia classification, thereby improving both accuracy and generalization[8]. However, prior VLPSO implementations mostly relied on static or time-varying parameter controls, without analyzing the generalization performance between training and testing data, thus limiting the practical reliability of VLPSO methods. Typically, VLPSO methods employ static or time-varying parameter controls without evaluating generalization between training and test data, leading to limited insights into real-world reliability. By optimizing the selection of relevant features, PSO helps improve classification accuracy, with reported results typically ranging from 80% to over 95% for SVM or KNN, depending on the dataset and classifier used[9], [10], [11]. This study proposes an improved VLPSO framework with two adaptive parameter control strategies, referred to as Adaptive1 and Adaptive2 parameter configuration, to optimize the control parameters, namely the inertia weight (w) and acceleration coefficient (c). The approach is validated through a 10-fold cross-validation using Leukemia gene expression datasets, with a focus on minimizing the train-test accuracy gaps across KNN, Decision Tree, and Naïve Bayes classifiers. The main contributions of this study are:

- Adaptive strategies were integrated into VLPSO for Leukemia diagnosis to enhance the algorithm's ability to adjust particle lengths during the optimization process.
- Sensitivity analysis of VLPSO key parameters (w and c) to optimize PSO performance. This helps avoid premature convergence and ensures a balance between global exploration and local exploitation, crucial for feature selection in high-dimensional cancer data.
- Finally, generalization between training and testing data will be evaluated to ensure that the proposed method delivers consistent classification accuracy across datasets and reliability for real-world applications.

2. Research Method

2.1 Literature Review Related to Research Method

PSO is one of the most well-regarded swarm-based algorithms in the literature, and a popular feature selection algorithm for GEP, wherein specific generated solutions randomly move within the search and work to obtain optimal solutions[12], [13]. Since Kennedy and Eberhart's initial development of PSO, researchers have proposed variants of this algorithm related to optimizing the solution[14], [15]. The fundamental concept of PSO involves learning from neighbors' experiences through communicating the global best (gbest) information and integrating each individual's own experience[16]. The velocity updating formula in PSO consists of three main components: w, which helps balance exploration and exploitation; the cognitive component ($c1$), which represents the particle's memory of its best position; and the social component ($c2$), which represents the influence of the swarm's best-known position. Although the original PSO has demonstrated good optimization performance, these two challenges lead PSO to suffer severely from premature convergence, requiring substantial computational resources and limiting flexibility in searching for more optimal feature subsets [17], [18]. Firstly, when w and c are not properly set, it directly affects the algorithm's ability to explore and exploit the search space effectively. This can lead to poor optimization results, especially in complex or high-dimensional problems. Secondly, Traditional PSO typically operates with a fixed-length population where each particle represents a solution of equal dimension. These limitations can result in suboptimal feature selection and reduced classification accuracy in leukemia classification.

A benchmark-constrained optimization problem was considered to study the sensitivity analysis of PSO parameter control by modifying each parameter individually, allowing particles to update their best historical solutions according to feasibility. The results of this research revealed that PSO was most sensitive to w (the cognitive component) and c (the social

component). Modifying the PSO control parameters can help researchers further enhance the performance of PSO[19]. Moreover, a detailed exploration of the interactions among these parameters can provide valuable insights into balancing global exploration and local exploitation, which is essential for optimizing performance. Studies have shown that adjusting these parameters effectively can enhance PSO's ability to avoid local optima and achieve more robust convergence toward global solutions [20]. Adaptive control of these parameters ensures that PSO maintains reliable performance across different datasets, enhancing its generalization capabilities. By tuning parameters to be responsive to the current state of the algorithm, PSO can produce consistent and reliable results, strengthening its overall reliability and applicability in real-world optimization tasks. The accuracy obtained from training and testing data often shows discrepancies; this can be attributed to overfitting during the training phase. Training results alone cannot be relied upon to assess the effectiveness of the analysis. Classification accuracy must also be monitored on test data. There will be differences between training and testing outcomes, where a minor difference is currently considered indicative of better generalization [21], [22]. Generalization analysis can be conducted by examining the difference between training and testing performance using tools such as box plots and heatmaps, which visually highlight the variability and consistency in the results.

The first modification introduced in PSO was the use of an inertia weight parameter in the initial PSO velocity update equation (wPSO), which consists of three classes: a constant value, a time-varying w strategy, and an adaptive strategy. The first class contains strategies where the value w is constant during the search or is determined randomly, commonly between 0.2 and 1.0. In a fast local search with gbest resetting PSO (PSO-LSRG), bare bones PSO (BBPSO), and a variable number of dimensions PSO (VNDPSO) share the same parameter settings c_1=c_2=2.0, with w linearly decreasing from 0.9 to 0.4 [16], [23]. The second class w is defined as a function of time referred to as a time-varying w strategy. These methods are not considered adaptive since they do not monitor the situation of the particles in the search space, which typically start at a higher value and gradually decrease over iterations, with a common strategy of w being 0.9 to 0.4. A multi-information fusion "triple variables with iteration" inertia weight PSO (MFTIWPSO) utilized a time-varying function w with [w_{max},w_{min}] = [0.5,1] [24]. In contrast, the linearly decreasing inertia weight (PSO-LDIW) was used with values of [0.9, 0.4] or [1.2, 0.9]. The random inertia weight PSO (PSO-RIW) was used [1,0.5] [20][25]. Variable length PSO (VLPSO) used c=1.49445 and time-varying w linear decrease in [0.9,0.2] that became the focus of this research. The third class is adaptive w strategies that use a feedback parameter to monitor the algorithm's state and adjust the inertia weight's value. Adaptive strategies often begin with an initial w value, similar to time-varying strategies, and are modified w based on feedback; this strategy could push w towards lower values. A chaotic inertia weight (PSO-CIW) sets the update velocity with c1 equals $c_1 = c_2 = 2.0$. It uses the adaptive w parameter determined by personal and global best locations [26]. The nonlinear inertia weight (PSO-NLIW) and an improved binary particle swarm optimization (IBPSO) enhanced c and w with linear fashion adaptation consisting of weighting inertia coefficient that computed and evaluated in each iteration to adjust the swarm finding the optimal location in the search space and improved the global search ability of the algorithm [27], [28], [29]. Since VLPSO currently employs a time-varying model, its integration of adaptive strategies represents a promising shift that could enhance its performance.

Researchers also state that the research on fixed-length optimization is mature; most PSO-based FS methods in the literature use the fixed-length representation, which has a length equal to the original number of population features and usually requires a significant amount of memory and computation time when applied to high-dimensional data. However, the research in variable-length optimization is still in its infancy [30]. The concept of variable-length chromosomes inspired VLPSO in Genetic Algorithms (GAs) and variable-length black hole optimization (VLBHO) with its two modes, position, and fitness, that permit flexible solution representation and allow particles to have adaptable lengths during the optimization process. The original VLPSO enables particles to have different and shorter lengths, leading to greater optimality and

https://doi.org/10.31849/digitalzone.v16i2. 27473

less computational load in VLPSO performance. Still, there was no separation in the treatment of training and test data, leaving a gap in understanding their impacts. This article also aims to study the sensitivity analysis of VLPSO using Leukemia cancer data in FS for early diagnosis and prognosis, to assess whether the generalization of experimental results is consistent between the training and testing data [31]. The remainder of the article is organized as follows. Section 2 presents the method. Afterward, the experimental results and discussion are presented in Section 3. Lastly, the conclusion and future works are presented in Section 4.

2.2 System Model and Problem Formulation

VLPSO (Variable-Length Particle Swarm Optimization) plays a crucial role in optimization tasks where flexibility in feature representation is essential, especially in high-dimensional datasets such as those found in cancer research. Sensitivity analysis in this context is used to understand how variations in key parameters, such as w and c, influence VLPSO's performance. By analyzing the sensitivity of these parameters, researchers can identify optimal settings that strike a balance between the algorithm's exploration and exploitation. This understanding ensures more consistent and reliable outcomes, which is essential for maintaining robust classification accuracy in medical data analysis. Integrating sensitivity analysis with VLPSO enables more adaptive and efficient parameter control, ultimately enhancing the algorithm's generalization and reliability in complex tasks, such as the classification of leukemia data.

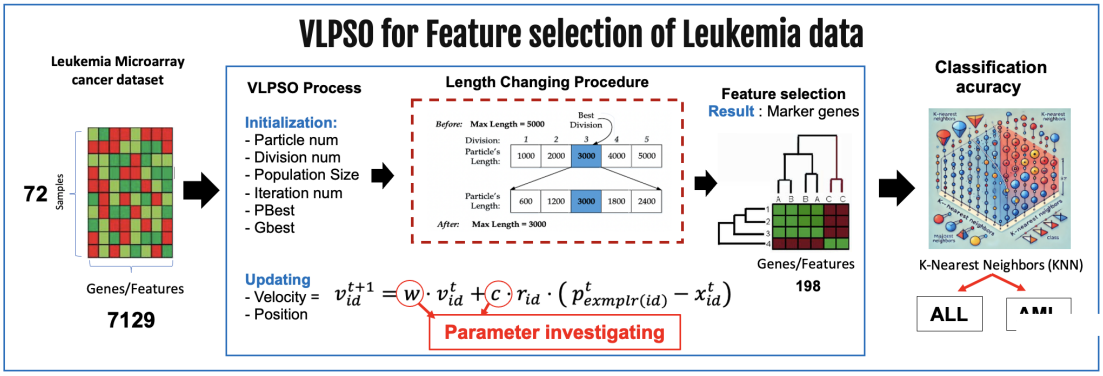


Figure 1. Research Framework for Parameter Setting in VLPSO for Leukemia Classification

Figure 1 illustrates the use of VLPSO for feature selection in leukemia data classification. The process begins with input data from a microarray dataset containing 72 samples, followed by 7,129 genes/features from the UCI Machine Learning repository at http://csse.szu.edu.cn/staff/zhuzx/Datasets.html. The challenge of analyzing this high-dimensional data is addressed by the VLPSO process, which starts with the initialization phase. This phase involves defining parameters such as the number of particles, division number, population size, and iteration count, as well as setting initial values for the personal best (*pBest*) and global best (*gBest*). The differentiation of the VLPSO algorithm from the original PSO lies in its improvements in particle length, feature ranking, and the dynamic adjustment of particle length during the feature selection process. Particles are not processed directly according to their full feature length, but are divided into divisions; each particle will have a different length based on this equation:

$$DivSize = \frac{PopSize}{NbrDiv} \quad (1)$$

$$ParLen_v = MaxLen * \frac{V}{NbrDiv} \quad (2)$$

The number of particles (or size) of each division (*DivSize*) is calculated based on the population size (*PopSize*) and the number of divisions (*NbrDiv*), as shown in (2). Particles in the same

division will have the same length. The length of particles in division *V* (*ParLenV*) is calculated based on (7), where the maximum length (MaxLen) is the dimensionality of the problem.

VLPSO incorporates the same parameters as CLPSO, enhancing the algorithm's effectiveness by improving the probability of particle selection, division allocation, and division length adjustment, ultimately leading to faster calculations and better performance[13]. The control parameters in CLPSO, specifically the inertia weight *w* and the acceleration constant *c,* ensure uniform influence in particle updates. The following equation governs the iterative update of particle velocity.

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \quad (3)$$

$$v_{id}^{t+1} = w_i \cdot v_{id}^{t} + c \cdot r_{id} \cdot pbest_{f_{i(d)}}^{t} - x_{id}^{t} \quad (4)$$

Where v_{id}^{t+1} represents the velocity of particle *i* at dimension *d,* and iteration *t*+1, and w_i is the inertia weight. This velocity equation enables particles to learn from the best personal positions of other particles, fostering diversity and robustness in the learning process. The update includes a random factor r_{id} in the range [0, 1], the best personal position $pbest_{f_{i(d)}}^{t}$ selected for particle *i* at dimension *d*, and the current position x_{id}^{t}. The position of a particle is subsequently updated by adding its new velocity. VLPSO employed specific static values for c = 1.49445 and used a time-varying type w, which influenced the search behavior over iterations using this updating w equation.

$$w = 0.9 - 0.5 * (\frac{current\ iteration}{iteration}) \quad (5)$$

For the next iteration, particle choosing is based on the probability of locating a better position or solution [26][27]. Particles should learn from particles with better fitness, and only those with better fitness should have a smaller Pc, allowing them to continue exploiting their good direction to find a better *pbest*. In contrast, the worst particles should learn from the better ones. The strategy adopted from comprehensive learning PSO (CLPSO) has shown promise in improving the performance of CLPSO for function optimization [32]. As shown in (6).

$$Pci = 0.05 + 0.45 \cdot \left\{ \frac{expexp\ (\frac{10(rank(i)-1)}{S-1})}{expexp\ (10)-1)} \right\} \quad (6)$$

To rearrange features in the descending order of their relevance, the symmetric uncertainty (SU), since it is a nonparametric measure and commonly used in FS methods, can be used to arrange the features [29]. SU is a normalized information gain (IG) version to evaluate feature relevance. To rank features, we use SU, as shown in (6) and (7), to measure the correlation between feature F and the class label C. The higher the feature correlates with the class label, the better it is

$$SU(F,C) = \left[\frac{IG(F|C)}{H(F)+H(C)} \right] \quad (7)$$

$$IG(C) = H(F) - H(C) \quad (8)$$

Where $H(F)$ represents the entropy of the feature, $H(C)$ is the entropy of the class, and $H(F|C)$ is the conditional entropy of the feature given the class. The higher the correlation measured SU, the more relevant the feature is deemed for classification purposes.

The process automatically changes the particles' length by cutting or appending more dimensions at the end of the representation while keeping the learned knowledge in the other dimensions. The number of dimensions being cut or appended is dynamically calculated based on the new length and the current length.

https://doi.org/10.31849/digitalzone.v16i2. 27473

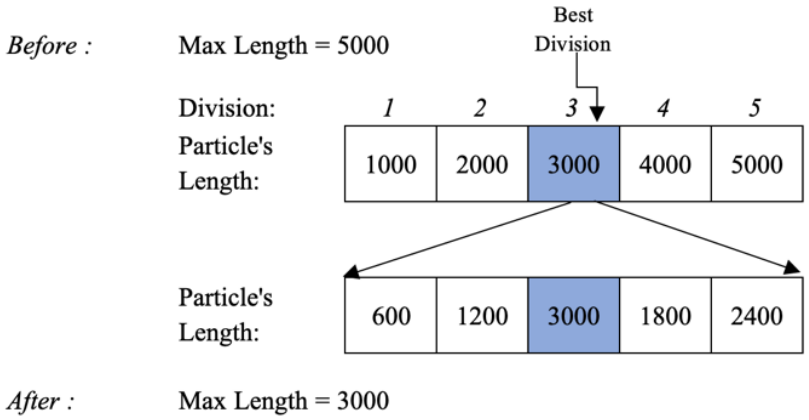


Figure 2. Example of length changing in a swarm with five divisions[33].

Figure 2 illustrates this process with a swarm comprising five divisions. Initially, the particle lengths of divisions 1, 2, 3, 4, and 5 are 1000, 2000, 3000, 4000, and 5000, respectively. Suppose the third division is the best division with the highest average fitness; it is kept unchanged, and 3000 becomes the new maximum length of the swarm. However, the particle length of divisions 1, 2, 4, and 5 will be changed to 600, 1200, 1800, and 2400, respectively. Therefore, the last 400, 800, 2200, and 2600 dimensions will be cut in particle representations of divisions 1, 2, 4, and 5, respectively.

These configurations are essential for ensuring that the algorithm can effectively navigate and explore the search space, adapting dynamically to optimize the feature selection process. The unique length-changing procedure embedded in VLPSO modifies the particle lengths throughout the search, thereby reducing the feature set to a more manageable size (e.g., 198 marker genes), which is crucial for building efficient and effective models. Figure 1 highlights the velocity update equation in VLPSO, which involves w and c. These parameters are a focal point of the research, aimed at determining their impact on the algorithm's ability to balance exploration and exploitation. In previous studies, VLPSO employed specific static values for c = 1.49445 and used a time-varying type for w, which influenced the search behavior over iterations. This research examines how various approaches, including adaptive strategies, can enhance the generalization and performance of VLPSO. The inertia weight w will be adjusted to an adaptive type, and this modification will be elaborated on in **Figure 3**. After the feature selection stage identifies marker genes, these selected features are fed into classification algorithms, such as K-Nearest Neighbors (KNN), Decision Trees, and Naïve Bayes, for data training and data testing to evaluate classification accuracy. The final step in the process involves determining the classification performance, with the goal of distinguishing between ALL and AML leukemia types.

2.3 Proposed Adaptive VLPSO Method

Figure 3 presents a comprehensive overview of the standard VLPSO procedure and introduces modifications for sensitivity analysis focused on parameter adjustments, specifically w, and c. The VLPSO standard procedure, as shown on the left side of the figure, includes initialization, velocity and position updates, feature ranking, pbest, and gbest updates. This setup establishes the foundation for feature selection by optimizing the swarm's movement within a defined search space. On the right side, new adaptive methods for updating w are shown, derived from various PSO algorithm strategies. The w value of VLPSO is adjusted using two adaptive equations derived from PSO variants: PSO Chaotic Inertia Weight (PSO-CIW) and Non-Linear Inertia Weight (PSO-NLIW)[19]. The original VLPSO algorithm uses a time-varying w parameter that decreases linearly according to equation (9):

$$w = 0{:}9 - 0{:}5 * \frac{current\ iteration}{Max\ iteration} \quad (9)$$

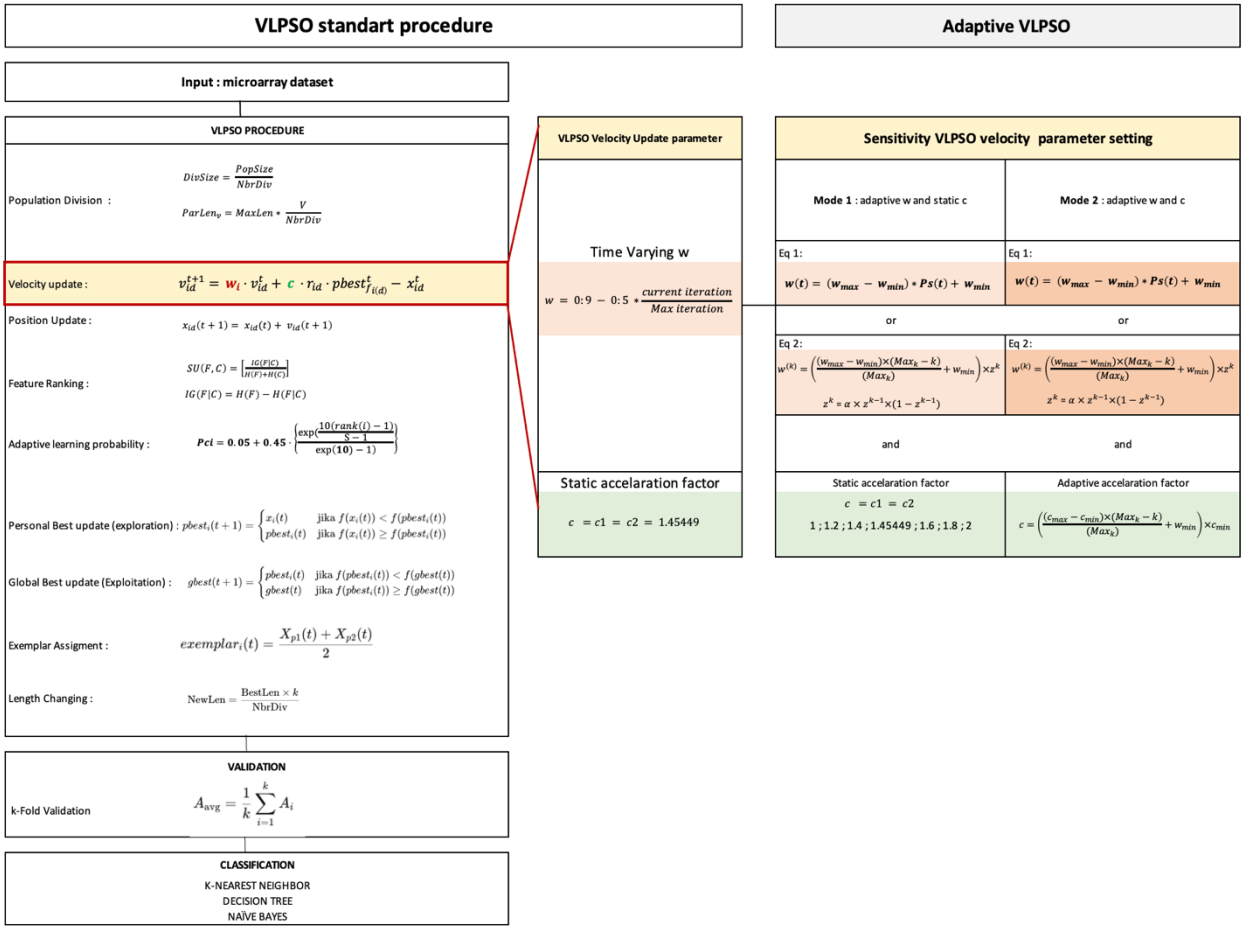


Figure 3. Research Design for Parameter Optimization in VLPSO

This linear decrease aims to balance exploration and exploitation throughout the optimization process. The PSO-CIW approach is modified w through an adaptive mechanism that incorporates chaotic sequences. Its equation is:

$$w(t) = (w_{max} - w_{min}) * Ps(t) + w_{min} \quad (10)$$

where $Ps(t)$ is a function of time that dynamically influences the weight, enabling the algorithm to explore the search space more effectively and avoid premature convergence; on the other hand, the PSO-NLIW method introduces a non-linear adaptation for w using the equation:

$$w^{(k)} = \left(\frac{(w_{max}-w_{min})\times(Max_k-k)}{(Max_k)} + w_{min}\right) \times z^k \quad (11)$$

$$z^k = \alpha \times z^{k-1} \times (1 - z^{k-1}) \quad (12)$$

Where z^k represents the current value of the chaotic variable at iteration k, α is a scaling factor or constant that influences the rate of change, or the magnitude of the adjustments applied to the variable z^k. This z^{k-1} is the value of the chaotic variable from the previous iteration. $1 - z^{k-1}$ is the term complement z^{k-1} by considering its inverse relationship within the update equation. It ensures that the product z^k is non-linear and introduces complexity to the adjustments, preventing repetitive or overly predictable changes. This non-linear weight adjustment considers

the state of the swarm and enhances the exploration capabilities while maintaining a balance with exploitation as the optimization progresses. These adaptive strategies have been proven in previous research to significantly improve the performance of PSO by adjusting w dynamically.

In PSO, the acceleration coefficient c plays a crucial role in balancing the exploration and exploitation of the search space. For standard VLPSO, the values of c_1, c_2, and c are set to 1.4495. This research examines the use of c values applied in previous PSO research, adapted for VLPSO, including c = *1, 1.2, 1.4, 1.4495, 1.6, 1.8,* and *2*. These modifications are incorporated to test whether adaptive adjustments to w and c can improve VLPSO's balance between global exploration and local exploitation. The c values to be used are selected based on the recommendations from previous studies that the sum of c1 and c2 for PSO should not exceed 4, such as those used in Binary Particle Swarm Optimization (BPSO), where $c_1 = c_2 = 2$[34], which is also applied in Hybrid Particle Swarm Optimization integrating a novel local search strategy (HPSO-LS)[35], Bare bones particle swarm optimization (BBPSO)[36], bare-bones PSO-based multi-objective evolutionary algorithm (BBMOPSO-A), and a fast "local search" combined with a gbest resetting mechanism (PSO-LSRG)[37]. The range of the value of c is obtained with a range of 0.9 to 1.90 for the variable number of dimensions in PSO (VNDPSO) [38], [39]. Random value with range [1.5,2.0] used in Crowding, mutation, and e-dominance PSO (CMDPSO), and the range [1.75,2.0] used for an improved binary particle swarm optimization (IBPSO) [38], [39]. The value of c 1.4495 is recommended from Internet protocol PSO (IPPSO), PSO with initialization pbest and gbest (PSOIniPG), non-dominating sorting PSO (NSPSOFS), and PSO Mutual Information (PSOMI) & PSO entropy (PSOE) [12], [28], [35], [36], [37], [40], [41]. It is important to note that the standard VLPSO also employs a static c value of 1.4495, serving as the standard reference for comparison. These are incorporated to test whether adaptive adjustments to w and c can improve VLPSO's balance between global exploration and local exploitation. The outcomes will be validated through K-fold cross-validation and tested with classifiers such as KNN, Decision Trees, and Naïve Bayes to evaluate the generalization and reliability of the performance on training and testing data. This investigation aims to demonstrate whether an adaptive w configuration contributes to enhanced performance in terms of reduced variance and improved reliability of classification results for VLPSO in early leukemia diagnosis.

3. Results and Discussion

The study aims to identify patterns of the parameter w to c determine which performs better compared to the standard VLPSO, which uses a time-varying type of w static $c = 1.49445$. A comparative analysis was conducted using boxplots and heatmaps to visualize and assess the stability, consistency, and generalization performance of each method.

3.1 Quantitative Evaluation of Generalization Performance

Table 1 below displays the training and testing accuracy values for the configurations of Adaptive1, Adaptive2, Time-varying, and Static for various values of static c. It also provides the difference between train and test accuracy (Train-Test) for each combination, which indicates the generalization performance of each method. A more minor Train-Test difference indicates better generalization. The count of the best Train-Test is at the bottom of the comparison table, indicating the number of times each method achieved the lowest Train-Test difference across the c values.

Table 1. Data training and testing differences in accuracy for KNN classification

c	Adaptive 1			Adaptive 2			Time Varying			Static		
	Train	Test	**Train-Test**	Train	Test	**Train-Test**	Train	Test	**Train-Test**	Train	Test	**Train-Test**
1	98.32	95.00	3.32	98.59	94.27	4.32	98.87	95.00	3.87	98.53	96.25	**2.28**
1.2	98.31	95.83	**2.48**	98.47	95.16	3.32	98.55	92.75	5.80	98.09	93.75	4.34
1.4	98.90	93.33	5.56	98.54	94.87	**3.67**	98.55	92.75	5.80	98.76	93.75	5.01
1.4495	98.99	93.75	5.24	98.53	94.85	3.69	99.22	95.00	4.22	99.11	96.25	**2.86**
1.6	98.66	94.58	4.07	98.49	94.93	3.56	98.43	93.58	4.84	98.64	96.25	**2.39**
1.8	98.54	94.58	3.96	98.29	94.65	**3.64**	98.21	92.08	6.13	98.09	93.75	4.34
2	99.12	95.00	4.12	98.57	94.85	**3.72**	98.24	92.13	6.11	99.56	95.25	4.31
Count of the best Train-Test			1			**3**			0			**3**

Based on **Table 1** above, for KNN classification, Adaptive1, Static, and Adaptive2 demonstrate the best balance between generalization and consistency. Static achieves the smallest Train-Test Difference at $c = 1$, and Adaptive2, maintaining stable performance across multiple c values. This indicates that these two models are more reliable for achieving consistent generalization across various parameter settings. Adaptive1, while showing strong generalization at one optimal point, $c = 1.2$ lacks consistency in its performance across the board. On the other hand, time-varying exhibits poor generalization capabilities, with larger Train-Test differences at all c values, suggesting that this model may not be suitable for tasks requiring strong generalization. Additionally, when examining the best Train-Test values, Adaptive2 and Static achieve the lowest difference in all three cases, reinforcing their reliability and consistent performance. Overall, for applications where consistency and robustness are crucial in KNN classification, Static and Adaptive2 appear to be the better choices among the options analyzed.

Table 2. Data training and testing differences in accuracy for the Decision Tree classification

c	Adaptive 1			Adaptive 2			Time Varying			Static		
	Train	Test	**Train-Test**	Train	Test	**Train-Test**	Train	Test	**Train-Test**	Train	Test	**Train-Test**
1	98.61	90.33	**8.28**	98.62	85.55	13.07	97.96	85.58	12.38	98.86	83.50	15.36
1.2	97.95	83.83	14.12	98.69	85.81	12.88	98.43	84.33	14.10	98.53	87.08	**11.44**
1.4	98.63	85.33	**13.30**	98.75	84.42	14.33	98.43	84.33	14.10	99.22	84.83	14.39
1.4495	98.53	89.08	**9.45**	98.63	85.65	12.98	98.51	85.83	12.67	98.62	85.17	13.45
1.6	98.30	89.08	9.22	98.65	84.77	13.89	98.44	90.92	**7.52**	98.87	83.33	15.54
1.8	98.43	88.83	**9.60**	98.63	84.93	13.70	98.40	86.83	11.57	98.53	87.08	11.44
2	98.53	86.83	**11.70**	98.75	84.40	14.34	98.42	86.68	11.74	98.63	84.58	14.05
Count of the best Train-Test			**5**						1			1

Based on **Table 2** above for Decision tree classification, Adaptive1 configuration demonstrates the best generalization performance, achieving the lowest Train-Test Differences in 5 out of 6 cases and dominating as the most robust model across different parameter settings, with its most minor difference at $c = 1$ (*i.e.*, Train-Test = 8.28), indicating strong and consistent generalization. Adaptive 2 shows moderate generalization, with its best difference at $c = 1.6$ (*i.e.*, Train-Test = 7.52) but is less consistent. It achieves only one best Train-Test count: Time-Varying shows limited performance, with only one competitive result at $c = 1.8$ (*i.e.*, Train-Test = 7.52). In contrast, Static consistently shows a larger Train-Test Difference, with its best at $c = 1.2$ (*i.e.*, Train-Test = 11.44). Adaptive1 is the preferred model for tasks needing reliable generalization and consistency for decision tree classification.

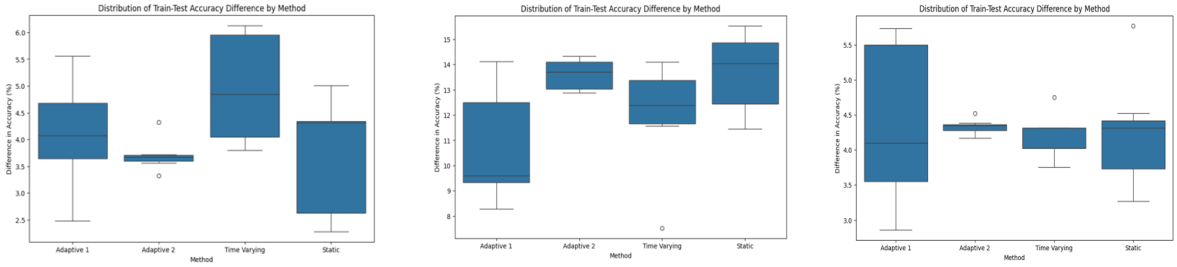
Table 3. Data training and testing differences in accuracy for Naïve Bayes classification

c	Adaptive 1			Adaptive 2			Time Varying			Static		
	Train	Test	**Train-Test**	Train	Test	**Train-Test**	Train	Test	**Train-Test**	Train	Test	**Train-Test**
1	99.11	96.25	**2.86**	99.56	95.03	4.52	100.00	95.25	4.75	99.44	95.25	4.19
1.2	99.66	96.50	**3.16**	99.53	95.30	4.23	100.00	96.25	3.75	99.77	96.50	3.27
1.4	99.55	94.00	5.55	99.57	95.25	4.32	100.00	96.25	**3.75**	99.31	95.00	4.31
1.4495	99.44	94.00	5.44	99.60	95.22	4.38	99.56	95.25	**4.31**	99.57	95.25	4.32
1.6	99.32	93.58	5.73	99.53	95.19	4.35	99.56	95.25	**4.31**	99.77	95.25	4.52
1.8	99.19	95.25	**3.94**	99.46	95.28	4.17	99.55	95.25	4.30	99.77	96.50	3.27
2	99.10	95.00	**4.10**	99.58	95.23	4.35	99.52	95.21	4.31	99.77	94.00	5.77
Count of the best Train-Test			**4**						**3**			**0**

Based on **Table 3**, Adaptive1 shows the strongest generalization capability, achieving the lowest Train-Test Differences in 4 out of 6 cases, with its best difference at $c = 1$ (*i.e.*, Train-Test = 2.86). This indicates that Adaptive1 is consistent and reliable for classification tasks requiring strong generalization. Adaptive 2 also performs well but is less consistent, with its best Train-Test Difference at $c = 1.8$ (*i.e.*, Train-Test = 4.17) and achieving the best results in 3 cases. Time-varying shows moderate performance with the three best Train-Test counts, indicating it can generalize effectively under certain conditions, with its best result at $c = 1.4$ (*i.e.*, Train-Test = 3.75). Static has only one count of the best Train-Test Difference at c=1.2 (*i.e.*, Train-Test = 3.27), suggesting that it is less reliable overall. For this Naïve Bayes Classification, the count of the best Train-Test highlights Adaptive 1's consistency, making it the most robust model, while Static lags in comparison.

3.2 Statistical and Variability Analysis Using Boxplots

Regarding the Quartile Range, median, mode, and average of the difference between train and test accuracy in the boxplot based on **Figure 4**, the comparison of the KNN, Decision Tree, and Naïve Bayes classifiers reveals distinct patterns in the performance of each method. These patterns provide insights into the variability and consistency of each method's generalization across different classifiers.



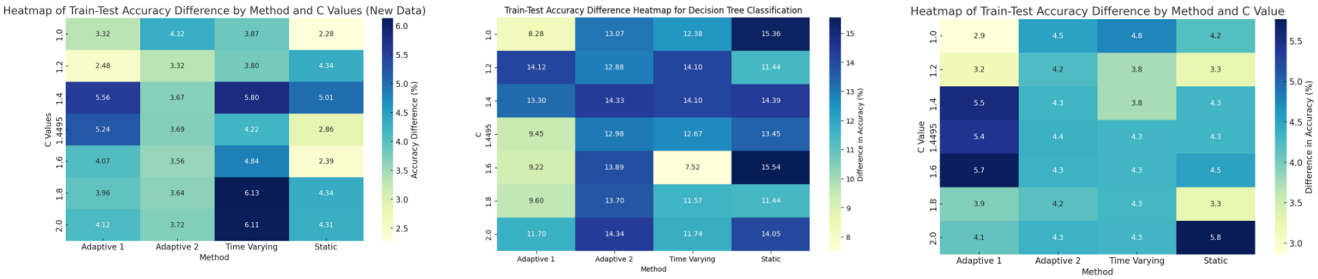
KNN Decision tree Naïve bayes

Figure 4. Boxplot comparison of all classifiers

Adaptive1 frequently shows the lowest median Train-Test accuracy differences in the boxplots, highlighting its potential to deliver better generalization performance under certain conditions. This indicates that Adaptive1 can effectively minimize both overfitting and underfitting, which is valuable for achieving optimal outcomes in specific applications. However, the wider interquartile range (IQR) and greater variability compared to the other methods suggest that Adaptive1 may not provide consistent reliability across all scenarios. In comparison, Adaptive2 demonstrates a more stable and narrower IQR with fewer outliers, indicating consistent performance, although its median Train-Test difference is often slightly higher than that of Adaptive1. Time-varying shows moderate performance with variability and occasional outliers, suggesting inconsistent generalization. In contrast, Static generally has the widest IQR and more outliers, indicating high variability and less dependable results. Overall, while Adaptive1 is strong for achieving the lowest differences when optimal conditions are met, Adaptive 2 stands out for its stability, making it preferable when consistent generalization is needed. Time-varying and Static exhibit more variability, making them less ideal for reliable performance.

3.3 Visualization of Generalization Trends Using Heatmaps

The heatmap analysis provides a comprehensive visualization of Train-Test accuracy differences for each method across various classifiers and c values. By examining these heatmaps, it becomes easier to identify patterns in generalization performance and pinpoint the conditions under which each method performs optimally. This section delves into the comparative configuration performance of Adaptive1, Adaptive2, Time-varying, and Static methods across the KNN, Decision Tree, and Naïve Bayes classifiers.



KNN Decision tree Naïve bayes

Figure 5. Heatmap comparison of all classifiers

Based on the heatmaps in **Figure 5** for KNN, Decision Tree, and Naïve Bayes, Adaptive1 generally shows the lowest Train-Test accuracy differences, for specific c values like $c = 1.2$ in KNN and $c = 1$ in Naïve Bayes, it indicates strong generalization under particular conditions.

3.1 Discussion

This finding aligns with [42], and [43], which showed that adaptive tuning methods can optimize classifier generalization by dynamically adjusting parameter sensitivity. Adaptive2 displays consistent moderate performance across all classifiers without major peaks or troughs, suggesting stable generalization but not consistently the best results, which is consistent with the stable convergence behavior reported in[44]. Meanwhile, time-varying strategies exhibit varied performance with occasional low accuracy differences, but also higher variability, echoing the instability noted in [45] for non-adaptive control strategies. Static methods perform less reliably overall, supporting prior findings that fixed parameters often fail to accommodate dataset-specific dynamics. Overall, Adaptive1 appears most effective in achieving low Train-Test differences when optimal c values are met; Adaptive2 is preferable for consistent and reliable generalization across different scenarios.

5. Conclusions

This study uses tables, boxplots, and heatmaps to compare the generalization, consistency, and variability of different VLPSO parameter settings for leukemia classification. The adaptive 1-parameter setting achieves the lowest Train-Test differences, especially with Decision Tree and Naïve Bayes, indicating strong generalization. However, its wider IQR shows higher variability. In contrast, the Adaptive2-parameter configuration demonstrates better consistency, with a narrower IQR and more stable performance across various settings. Time-varying methods show moderate, less consistent outcomes, while static methods perform the worst. These findings suggest that adaptive tuning for VLPSO parameters (w and c) enhances both generalization and stability. Such improvements are valuable for designing robust and accurate medical classification systems, especially in bioinformatics and clinical decision support tools.

The heat maps further emphasize that the Adaptive1 parameter setting is effective under specific conditions and achieves the lowest Train-Test differences. At the same time, the Adaptive2 is reliable for maintaining stability across different scenarios. These findings suggest that adaptive parameter settings, such as those used in Adaptive 1, are beneficial for optimizing VLPSO, as they improve performance and generalization under varying conditions. It is recommended to use adaptive parameter tuning for w and c VLPSO to achieve better overall consistency and generalization, balancing flexibility and stability in optimization tasks.

References

[1] J. M. Castillo, M. Arif, W. J. Niessen, I. G. Schoots, and J. F. Veenland, "Automated Classification of Significant Prostate Cancer on MRI: A Systematic Review on the

Performance of Machine Learning Applications." *Cancers,* vol. 12, no. 6, pp. 1-13, 2022, doi: https://doi.org/10.3390/cancers12061606

[2] R. Alfano, G. S. Bauman, J. A. Gomez, M. Gaed, M. Mousa, J. Chin, et al, "Prostate cancer classification using radiomics and machine learning on mp-MRI validated using co-registered histology," *European Journal of Radiology*, vol. 156, no. 110494, pp. 1-9, 2022, doi: https://doi.org/10.1016/j.ejrad.2022.110494

[3] M. A. Mazurowski, M. Buda, A. Saha, and M. R. Bashir, "Deep Learning in Radiology: An Overview of the Concepts and a Survey of the State of the Art with Focus on MRI," *Journal of Magnetic Resonance Imaging*, vol. 49, no. 4, pp. 939–954, 2018, doi: 10.1002/jmri.. 26534.

[5] T. R. Mahesh, D. Santhakumar, A. Balajee, H. S. Shreenidhi, V. V. Kumar, and J. Rajkumar Annand, "Hybrid Ant Lion Mutated Ant Colony Optimizer Technique With Particle Swarm Optimization for Leukemia Prediction Using Microarray Gene Data," *IEEE Access*, vol. 12, no. December 2023, pp. 10910–10919, 2024, doi: 10.1109/ACCESS.2024.3351871.

[6] L. Alzubaidi *et al.*, "Towards a better understanding of transfer learning for medical imaging: A case study," *Applied Sciences (Switzerland)*, vol. 10, no. 13, pp. 1–21, 2020, doi: 10.3390/app10134523.

[7] D. Santhakumar and S. Logeswari, "Efficient attribute selection technique for leukaemia prediction using microarray gene data," *Soft Comput*, vol. 24, no. 18, pp. 14265–14274, 2020, doi: 10.1007/s00500-020-04793-z.

[8] B. Tran, B. Xue, and M. Zhang, "Variable-Length Particle Swarm Optimization for Feature Selection on High-Dimensional Classification," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 3, pp. 473–487, 2019, doi: 10.1109/tevc.2018.2869405.

[9] K. Hamad, R. M. Ahmed, and B. Celik, "Using Machine Learning to early detection and classification of breast cancer masses based on medical image processing." *Research Square*, vol. 1, pp. 1-22, 2023, doi: https://doi.org/10.21203/rs.3.rs-2973046/v1

[10] K. Sangeetha and S. Prakash, "An Early Breast Cancer Detection System Using Stacked Auto Encoder Deep Neural Network with Particle Swarm Optimization Based Classification Method," *J Med Imaging Health Inform*, vol. 11, no. 12, pp. 2897–2906, 2021, doi: 10.1166/jmihi.2021.3886.

[11] J. M. C. T. Veenland, M. Arif, W. J. Niessen, I. G. Schoots, and J. F. Veenland, "Automated classification of significant prostate cancer on MRI: A systematic review on the performance of machine learning applications," *Cancers (Basel)*, vol. 12, no. 6, p. 1606, 2020, doi: 10.3390/cancers12061606

[12] T. M. Shami, A. A. El-Saleh, M. Alswaitti, Q. Al-Tashi, M. A. Summakieh, and S. Mirjalili, "Particle Swarm Optimization: A Comprehensive Survey," *IEEE Access*, vol. 10, pp. 10031–10061, 2022, doi: 10.1109/ACCESS.2022.3142859.

[13] N. S. Mohammed and H. Beitollahi, "Accurate Classification in Uncertainty Dataset Using Particle Swarm Optimization-trained Radial Basis Function," *SSRN Electronic Journal*, vol. 20, no. 9, pp. 1-14, 2022, doi: 10.2139/ssrn.4203377.

[14] J. Kennedy, "Particle swarm: Social adaptation of knowledge," in *Proceedings of the IEEE Conference on Evolutionary Computation, ICEC*, 1997, pp. 303–308. doi: 10.1109/icec.1997.592326.

[15] J. Kennedy and R. C. Eberhart, "Discrete binary version of the particle swarm algorithm," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, 1997, pp. 4104–4108. doi: 10.1109/icsmc.1997.637339.

[16] B. Tran, B. Xue, M. Zhang, and S. Nguyen, "Investigation on particle swarm optimisation for feature selection on high-dimensional data: local search and selection bias," *Conn Sci*, vol. 28, no. 3, pp. 270–294, 2016, doi: 10.1080/09540091.2016.1185392.

[17] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," *Inf Process Lett*, vol. 85, no. 6, pp. 317–325, 2003, doi: 10.1016/S0020-0190(02)00447-7.

[18] S. Jingqaio, Zahng; Arthur C., "JADE: Self-Adaptive Differential Evolution with Fast and Reliable Convergence Performance," *Evol Comput*, vol. 13, no. 5, pp. 2251–2258, 2007. https://dl.acm.org/doi/abs/10.1109/tevc.2009.2014613?utm

[19] M. Isiet and M. Gadala, "Sensitivity analysis of control parameters in particle swarm optimization," *J Comput Sci*, vol. 41, pp. 101086, 2020, doi: 10.1016/j.jocs.2020.101086.

[20] M. Jain, V. Saihjpal, N. Singh, and S. B. Singh, "An Overview of Variants and Advancements of PSO Algorithm," vol. 12, no.17, pp. 1-21 2022, *MDPI*. Doi: 10.3390/app12178392.

[21] E. K. Ampomah, Z. Qin, and G. Nyame, "Evaluation of tree-based ensemble machine learning models in predicting stock price direction of movement," *Information (Switzerland)*, vol. 11, no. 6, pp. 1-22, 2020, doi: 10.3390/info11060332.

[22] S. Camalan *et al.*, "Convolutional neural network-based clinical predictors of oral dysplasia: Class activation map analysis of deep learning results," *Cancers (Basel)*, vol. 13, no. 6, pp. 1–18, Mar. 2021, doi: 10.3390/cancers13061291.

[23] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004, doi: 10.1109/TEVC.2004.826071.

[24] M. Li, H. Chen, X. Shi, S. Liu, M. Zhang, and S. Lu, "A multi-information fusion `triple variables with iteration' inertia weight PSO algorithm and its application," *Appl. Soft Comput.*, vol. 84, no. Nov, p. 105677, 2019. doi: 10.1016/j.asoc.2019.105677

[25] D. Cao, Y. Xu, Z. Yang, H. Dong, and X. Li, "An enhanced whale optimization algorithm with improved dynamic opposite learning and adaptive inertia weight strategy," *Complex & Intelligent Systems*, vol. 9, no. 1, pp. 767–795, 2023, doi: 10.1007/s40747-022-00827-1.

[26] S. Ajibade *et al.*, "An Insight Into The Modification Of Inertia Weight Of Pso For Feature Selection," *Article in Journal of Biomechanical Science and Engineering*, vol. July, pp. 683-700, 2023, doi: 10.17605/OSF.IO/R2ZUF.

[27] A. T. Azar, Z. I. Khan, S. U. Amin, and K. M. Fouad, "Hybrid Global Optimization Algorithm for Feature Selection," *Computers, Materials and Continua*, vol. 74, no. 1, pp. 2021–2037, 2023, doi: 10.32604/cmc.2023.032183.

[28] B. Ji, X. Lu, G. Sun, W. Zhang, J. Li, and Y. Xiao, "Bio-Inspired Feature Selection: An Improved Binary Particle Swarm Optimization Approach," *IEEE Access*, vol. 8, pp. 85989–86002, 2020, doi: 10.1109/ACCESS.2020.2992752.

[29] B. Nouri-Moghaddam, M. Ghazanfari, and M. Fathian, *"*A novel bio-inspired hybrid multi-filter wrapper gene selection method with ensemble classifier for microarray data*."* *Springer Nature*, vol. 35, no. 16, pp. 11531-11561, 2023. doi: 10.1007/s00521-021-06459-9.

[30] T. O. Q. Saraf, N. Fuad, and N. S. A. M. Taujuddin, "Framework of Meta-Heuristic Variable Length Searching for Feature Selection in High-Dimensional Data," *Computers*, vol. 12, no. 1, pp. 1–13, 2023, doi: 10.3390/computers12010007.

[31] B. Tran, B. Xue, and M. Zhang, "Variable-Length Particle Swarm Optimization for Feature Selection on High-Dimensional Classification," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 3, pp. 473–487, 2019, doi: 10.1109/tevc. 2018.2869405.

[32] Y. Cao, H. Zhang, W. Li, M. Zhou, Y. Zhang, and W. A. Chaovalitwongse, "Comprehensive Learning Particle Swarm Optimization Algorithm with Local Search for Multimodal Functions," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 4, pp. 718–731, 2019, doi: 10.1109/TEVC.2018.2885075.

[33] B. Tran, B. Xue, and M. Zhang, "Variable-Length Particle Swarm Optimization for Feature Selection on High-Dimensional Classification," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 3, pp. 473–487, 2019, doi: 10.1109/TEVC.2018.2869405.

[34] S. Sazzed, "ANOVA-SRC-BPSO: a hybrid filter and swarm optimization-based method for gene selection and cancer classification using gene expression profiles," *Proceedings of*

the Canadian Conference on Artificial Intelligence, pp. 1-12. 2021, doi: 10.21428/594757db.9e9e0337.

[35] P. Moradi and M. Gholampour, "'A hybrid particle swarm optimization for feature subset selection by integrating a novel local search strategy,'" *Applied Soft Computing Journal*, vol. 43, no. C, pp. 117–130, 2016, doi: 10.1016/j.asoc.2016.01.044.

[36] C. Qiu, "Bare bones particle swarm optimization with adaptive chaotic jump for feature selection in classification," *International Journal of Computational Intelligence Systems*, vol. 11, no. 1, pp. 1–14, 2018, doi: 10.2991/ijcis.11.1.1.

[37] L. Y. Chuang, C. S. Yang, K. C. Wu, and C. H. Yang, "Gene selection and classification using Taguchi chaotic binary particle swarm optimization," *Expert Syst Appl*, vol. 38, no. 10, pp. 13367–13377, 2011, doi: 10.1016/j.eswa.2011.04.165.

[38] P. Kadlec, "Multi-Objective PSO with Variable Number of Dimensions for Space Robot Path Optimization," *Algorithms*, vol. 16, no. 6, 2023, doi: 10.3390/a16060307.

[39] Y. O. M. Sekyere, F. B. Effah, and P. Y. Okyere, "An Enhanced Particle Swarm Optimization Algorithm via Adaptive Dynamic Inertia Weight and Acceleration Coefficients," *Journal of Electronics and Electrical Engineering*, vol. 3, no. 3, pp. 53-67, doi: 10.37256/jeee. 3120243868.

[40] B. Xue, M. Zhang, and W. N. Browne, "Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms," *Applied Soft Computing Journal*, vol. 18, pp. 261–276, 2014, doi: 10.1016/j.asoc.2013.09.018.

[41] X. F. Song, Y. Zhang, Y. N. Guo, X. Y. Sun, and Y. L. Wang, "Variable-Size Cooperative Coevolutionary Particle Swarm Optimization for Feature Selection on High-Dimensional Data," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 5, pp. 882–895, 2020, doi: 10.1109/TEVC.2020.2968743.

[42] K. Tadist, S. Najah, N. S. Nikolov, F. Mrabti, and A. Zahi, "Feature selection methods and genomic big data: a systematic review," *J Big Data*, vol. 6, no. 1, pp. 1–4, 2019, doi: 10.1186/s40537-019-0241-0.

[43] M. Charikar, C. Pabbaraju, and K. Shiragur, "Quantifying the Gain in Weak-to-Strong Generalization," in *38th Conference on Neural Information Processing Systems (NeurIPS 2024)*, 2024, pp. 1–26, available: https://proceedings.neurips.cc/paper_files/paper/2024/hash/e4a0d8aef3567f742b0794844d9b5847-Abstract-Conference.html

[44] C. Burns *et al.*, "Weak-to-Strong Generalization: Eliciting Strong Capabilities With Weak Supervision," *ArXiv*, vol. 1, pp. 1–49, 2023, doi: http://arxiv.org/abs/2312.09390

[45] S. Ramadhani *et al.*, "Feature Selection Optimisation for Cancer Classification Based on Evolutionary Algorithms: An Extensive Review," *Computer Modeling in Engineering & Sciences*, vol. 143, no. 1, pp. 2712–2765, 2025, doi: 10.32604/cmes . 2025.062709.