Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.060765

REVIEW

# Particle Swarm Optimization: Advances, Applications, and Experimental Insights

**Laith Abualigah***

Computer Science Department, Al al-Bayt University, Mafraq, 25113, Jordan

*Corresponding Author: Laith Abualigah. Email: aligah.2020@gmail.com

Received: 09 November 2024; Accepted: 26 December 2024; Published: 17 February 2025

**ABSTRACT:** Particle Swarm Optimization (PSO) has been utilized as a useful tool for solving intricate optimization problems for various applications in different fields. This paper attempts to carry out an update on PSO and gives a review of its recent developments and applications, but also provides arguments for its efficacy in resolving optimization problems in comparison with other algorithms. Covering six strategic areas, which include Data Mining, Machine Learning, Engineering Design, Energy Systems, Healthcare, and Robotics, the study demonstrates the versatility and effectiveness of the PSO. Experimental results are, however, used to show the strong and weak parts of PSO, and performance results are included in tables for ease of comparison. The results stress PSO's efficiency in providing optimal solutions but also show that there are aspects that need to be improved through combination with algorithms or tuning to the parameters of the method. The review of the advantages and limitations of PSO is intended to provide academics and practitioners with a well-rounded view of the methods of employing such a tool most effectively and to encourage optimized designs of PSO in solving theoretical and practical problems in the future.

**KEYWORDS:** Particle swarm optimization (PSO); optimization algorithms; data mining; machine learning; engineering design; energy systems; healthcare applications; robotics; comparative analysis; algorithm performance evaluation

## 1 Introduction

The process of optimization is crucial for solving various intricate decision-making tasks in engineering, medicine, finance, and artificial intelligence, among other fields [1]. With the increasing complexity of systems and datasets, there is an ever-increasing demand for accurate and effective optimization methods [2,3]. Optimization problems are defined as problems of selecting the "best" of several alternatives characterized by feasible regions and some specified constraints, with a focus on one or more likely functions [4,5]. A specific context is crucial since it defines which goal is given priority: cost minimization, efficiency maximization, accuracy enhancement, or trade-off between some conflicting goals. Such problems arise in many real-world situations, such as energy system design, forecasting, manufacturing, and intelligent decision support systems [6,7]. Nowadays, the increasing complexity of these applications has increased the need for optimization for methods that are both applicable and plausible in real-life settings [5].

Finding optimal solutions can be particularly difficult considering the vast complexity that exists in real-world issues or concerns, especially those that have large and high dimensional characteristics. As the problem sizes increase, computation times, as well as the effort required to look for optimal solutions, become too demanding. Therefore, traditional methods such as linear programming and exhaustive search approaches might not be sufficient. Additionally, many classic methods tend to handle problems that are non-linear, non-differentiable, and multi-modal quite poorly. For instance, real-life problems tend to include a multitude of locally optimal solutions broader than what is considered the globally optimal solution, which makes reaching it quite difficult. Therefore, the constraints that conventional techniques tend to have led to the optimization sector being more focused on heuristic and metaheuristic algorithms. These algorithms focus on achieving a result that is closest to the best result without searching all possible solutions. Instead of providing a search that is in as much as all possible outcomes, these algorithms can provide accurate estimates.

Many advantages are presented using heuristic and metaheuristic algorithms, especially for complex optimization tasks [8]. They are not as rigidly defined as classical ones, and this is an advantage in cases when the problem cannot be precisely described mathematically or when no gradient information is available. Metaheuristics, in particular, seem to provide a heuristic based on prior experiences that deliver quick computation on high-quality outcomes [9,10]. Among these techniques, Particle Swarm Optimization (PSO) has gained immense popularity owing to its ease of use, versatility and the ability to solve diverse optimization problems over many areas. PSO was first introduced by Kennedy and Eberhart in 1995 [11,12]. The PSO is based on the concepts of social behavior found in a group of birds fishes, or insects. The strategy of the algorithm is to utilize this group intelligence to seek the best solution in a complicated one-dimensional domain, where each of the agents, that is referred to as particles, changes its direction based on personal and social experiences to navigate through the search domain.

PSO utilizes a swarm of particles, each of which specifies a possible solution [13]. These particles "fly" in the search space and change their locations based on their best-found position and the position best known to their neighbors. Through this cooperative mechanism, PSO can effectively perform both, exploration and exploitation—exploration by enabling particles to different areas without allowing them to converge early and exploitation by concentrating the search in the perspective areas to improve the search. This balance is essential to encompass the complicated search spaces and find solutions efficiently. By virtue of the particle-based, decentralized mechanism of PSO, it can change dynamically for most of the problem landscape, and that is why it works well in the areas of multi-dimensional and continuous practices of optimization [14].

Ever since it appeared, PSO has become a household name in research and on the industrial front due to its fast convergence rate and uncomplicated parameter setting as compared to other metaheuristics such as Genetic Algorithms (GA) [15] or Simulated Annealing (SA) [16]. PSO has, by virtue of its structure, a straight-forward implementation process and a low computational burden, which makes it ideal for circumstances in which there are resource constraints [17]. Furthermore, the PSO's robustness has allowed users to customize its basic architecture and further improve and optimize the model for more specific applications [18]. Such modifications are usually aimed at changing an evolutionary search strategy, increasing convergence rate, or avoiding premature convergence to local optima, which usually constitute major limitations whenever the model is used on some complex problems [19,20].

However, the PSO standard has some drawbacks that should be addressed before deploying it in real-world applications, like high dimensional complex problems, problems involving more than one contradictory objective, etc. For Example, the algorithm configuration is prone to parameter sensitivity, which can affect its performance, and it sometimes needs to properly explore the parameter space. Such limitations have led researchers to explore various improvements and combinations of PSO with the aim of enhancing its rate of convergence, robustness, and adaptability. Of late, new horizons have been opened for the development of PSO, such as tandem PSO variants that incorporate other optimization methods, adaptive PSO algorithms applying dynamic parameter tuning, and parallelized versions that utilize distributed processing to enhance computerized efficiency. These have broadened the range of applicability of PSO and made it possible to solve difficult problems in the areas of engineering design optimization, energy management, machine learning, medicine, etc.

As such, PSO remains a helpful method of optimization as it improves and adjusts to the requirements of contemporary applications. Its flexibility has brought about remarkable results, particularly in areas where the problem landscapes are so intricate, and the traditional methods of optimization are inadequate. The growing use of PSO in cross-disciplinary studies indicates its success in applying a pragmatic approach to solve some of the most difficult optimization problems encountered today.

This research paper aims to tackle particle swarm optimization from a modern perspective and describe its other important aspects, including recent advancements and applications practiced in various areas. The specific goals are:

- To survey the principles of PSO and mention some of the recent changes, which, in a way, build upon its quite a few historical shortcomings.
- To describe and evaluate the use of PSO in the following six areas: Data Mining, Machine Learning, Engineering Design Optimization, Energy Systems, Healthcare, and Robotics. Each of them has its peculiarities in terms of what needs optimal solutions and how the algorithm's (PSO in this case) effectiveness is measured.
- This paper provides a relative sense of PSO's effectiveness by comparing it with other popular optimization methods like Genetic Algorithms, Ant Colony Optimization, and Differential Evolution. This comparison seeks to highlight PSO's strengths and weaknesses against the backdrop of similar algorithms and suggest scenarios in which PSO can be the best option.
- This paper aims to inform the particle swarm optimization community, among others, about the strengths and weaknesses of PSO so that researchers and practitioners can easily determine when and how best to use the algorithm in practice.

The paper is structured in the following way:

In Section 2, we review the basic concepts, processes, and recent developments of Particle Swarm Optimization (PSO), highlighting its adaptability to various optimization problems. Section 3 compares PSO's performance with other optimization techniques through case studies, showcasing its competitive advantages. Section 4 explores six key application areas—Data Mining, Machine Learning, Engineering Design Optimization, Energy Systems, Healthcare, and Robotics—supported by experimental results and comparative analysis. Section 5 examines PSO's advantages and limitations in global optimization and assesses the impact of recent advancements. Section 6 outlines future research directions, including hybridization, adaptive parameterization, and new applications, concluding with recommendations for researchers and practitioners.

## 2 Overview of Particle Swarm Optimization (PSO)

### *2.1 Fundamentals of PSO: Algorithm Principles, Key Components, and Parameters*

Particle Swarm Optimization (PSO), an evolution of population-based algorithms, mimics the flocking behaviors of birds, schools of fish, or even insect swarms. Kennedy and Eberhart developed it in 1995; PSO uses a vast area of search space filled with particles or "agents" that act cohesively with the problem's search space to reach an optimization solution [18]. Each category of particles gives possible solutions to the problem. The category of particles enhances its own experience determined by the movement of other shares or parts through the swims. Particles enhance their cooperation, increasing the chances of optimal solutions efficiently and rapidly depending on the circumstances [21].

PSO itself is notable for its ease of implementation (which takes just a few code lines) and efficiency. It is applicable in engineering, machine learning, data triage, and beyond [22,23]. This method is becoming more common to tackle optimization instead of conventional approaches mathematically stiff for a gradient due to nonlinear multidimensional problems. PSO is constrained without the use of gradient information transfer, hence operating in environments of non-differentiable multimodal and noisy search space.

a) The following are the main principles that guide the application of the PSO algorithm:

- Social Information Sharing Strategy: In the application of the PSO algorithm, particles relay information with their counterparts on locating promising spots in the search space algorithms thus allowing the swarm to make travel towards the best solutions [24,25].
- Exploration *vs.* Exploitation Balance Optimization: The algorithm takes care of exploration—searching new regions of space—and exploitation—exploiting already found good regions. The components of inertia, cognitive and social, progressively focus on this bias and allow the swarm to explore a large search space. Still, at the same time, they concentrate on high-fitness solutions.
- Adaptations Based on Individual and Global Experience: The movement of particles is determined by the individual's best position and the best global position that the swarm could reach, making the system's search pattern flexible but firmly directed.

b) Main Formula and Elements

In the context of PSO, a particle is defined as a candidate solution and has spatial coordinates and velocity in the solution space. Let us list some of the most important components together with their mathematical representations of the Particle Position and Velocity:

- Position $(x\_i, t)$: Every particle in the search space is a likely solution to the optimization problem for that particular component.
- Velocity $(v\_i, t)$: Velocity defines how fast and in what direction the particle will move through the search space at the timestamp $t$.
- Velocity Update Equation: Three major components are the basis for the velocity update of every particle:
- Inertia Component: The term $\omega * (v\_i, t)$ states the past tendency of a particle with respect to the previous velocity, where ω (inertia weight) modifies whether or not the movement in previous instances affects the current direction.
- Cognitive Component: This controlled factor, $c1 * (r1, i) * (p\_i - x(i, t))$, aims the particle to move towards its best fitness position. In this case, c1 is termed the cognitive coefficient, and $r1$, $i$ is a random number generated in a distribution of [0, 1] where $p\_i$ is the best position known by the particle
- Social Component: This controlled factor, $c2 * (r2, i) * (g - x(i, t))$, directs the particle to the global position identified by any particle within the swarm. Here, $c2$ is termed the social coefficient, while $(r2, i)$, and $g$ are a random variable and global best position, respectively.

The following equation does the update of the velocity:

$$(v\_i, t) + 1 = \omega * (v\_i, t) + c1 * r1, i * (p\_i - x(i, t)) + c2 * r2, i * (g - x(i, t)) \tag{1}$$

where, past cognition or inertia, personal experience or cognitive component, and social factors and social component together govern the velocity of the particle.

Position Update Equation: Each particle's movement across the search space is performed by adding the new location or updated position, which is achieved by adding the new velocity to the particle's current position.

$$(x\_i, t+1) = (x(i,t)) + (v\_i, t) + 1 \tag{2}$$

c) Control Parameters

The performance and behavior of PSO are influenced by several control parameters few of which are detailed below [26]:

- Inertia Weight ($\omega$): The inertia weight governs the relative influence of the particle's prior motion on the current motion. A greater value of ω enables exploration through inertia, while a smaller value of ω promotes exploration by reducing the speed near attractive zones. Generally, throughout the search process, ω changes in order to complement the exploration and the exploitation.
- Cognitive and Social Coefficients ($c1$ and $c2$): Cognitive coefficient c1 measures the extent to which a particle applies knowledge from its own experience, while social coefficient $c2$ measures the extent to which $c1$ applies to the best position in the swarm. By varying $c1$ and $c2$, it is possible to set the algorithm to pursue social sharing or individual learning instead.
- Randomness ($r1, i$ and $r2, i$): Random variables $r1, i$ and $r2, i$ sampled from uniform distribution in $[0,1]$ ensure variability in the search process so as to avoid premature convergence and also to enhance a wider search of the search space.

d) Working Steps of PSO

The PSO functions in a series of steps, which are:

- Initialization: Randomly assign positions and velocities to the particles in the swarms within the search space. The system also fixes all the particles' best-known positions, i.e., the position, $p\_i$ for each particle and the most qualified position $g$ according to the zero fitness evaluation of the members of the particle swarm.
- Fitness Evaluation: For every iteration, carry out the corresponding fitness evaluation of the position occupied within the current step (for the respective particles). This will depend on the optimization features and form an evaluation mark for the quality of the solution provided.
- Update the Personal Bests and Global Bests: For each particle, find out whether the current position of the flock is better than the already known best position $p\_i$ of the particular particle. If that is the case, update the best *position* $p\_i$. If any other particle of the swarm achieves a lesser-known best fitness value, update the already known best fitness $g$.
- Update Velocity and Position: For every particle, substituting the value into the velocity update formula gives $(v\_i, t+1)$. A new position is given by $(x(i,t)+1) = x(i,t) + (v\_i, t+1)$ on adding the updated velocity to the previous position $(x(i,t))$.
- Iteration and Termination: Continue repeating fitness evaluation, and the update steps for a specified predefined number of iterations or particular criteria for terminating are attained (for example, a marginally satisfactory fitness measure and a convergence level).
- Your understanding of the tasks and functions required for input optimization is truly commendable. As soon as we are able to finalize the details concerning the unification equation, we can undertake a series of tests focused on determining the performance parameters noted in the output section.

e) PSO Variants and Enhancements

To enhance the efficacy of PSO techniques and tackle the barriers of early saturation, local diving or more optimal solutions, several variants and improvements were made to the original Particle Swarm Algorithm. A few of those improvements are:

- Adaptive Inertia Weight Adjustment: Changing the value of $\omega$ according to the iteration or the swarm activities in order to increase the balance between explorative and exploitative aspects of particle swarms.
- Hybrid PSO Models: PSO fused with other dominant paradigms for optimization, such as GA or Differential Evolution, to apply more than one strategy of searching for the optimum.
- Multi-Swarm Approaches: More than one swarm is employed to ensure that the various regions of the search area are worked on, which improves robustness in complex landscapes and minimizes chances of premature convergence.

## 2.2 Recent Advancements in PSO

The past two decades have witnessed a veritable explosion in work on PSO with both spawned in the mid-90s as powerful optimization algorithms that have boosted metaheuristics [27]. Building on over two decades of research, technical papers have popularized and developed countless variants of both algorithms across the globe. Nonetheless, the literature has not provided a balanced assessment of the variants of PSO and DE. The present work fills this void by testing a total of 20 variants, 10 featuring PSO and the other 10 featuring DE, including the most recent ones from 2022 through experimental benchmarks with early versions on 25 artificial and real-world problems. It was deduced that, in most of the cases, DE was the best approach, at least relative to PSO, which seems not to be proportionately large in literature where it is cited two to three times more than DE, as one example noted. Some problems where it appears that PSO has the advantage over DE, such cases appear to be few and far between. Perhaps the most important practical implication is that there is a need to rethink a number of algorithmic design principles that underpin the PSO paradigm.

PSO, a social-inspired stochastic and population-based algorithm, has been adapted widely to tackle water resource optimization problems. This study critically reviews the basic principles of PSO search algorithms [28]. It focuses in detail on the applications and performance of PSO in engineering-solving tasks like the operation of reservoirs, rainfall-runoff and water quality modeling, and groundwater modeling. A systematic literature review identified 22 distinct PSO variants, all of which met the defined challenges in dealing with specific water resource issues. The study evaluates the performance of PSO variants against other EAs and mathematical programming, including simulated annealing, differential evolution, genetic algorithms, shark algorithms support vector machine, and differential dynamic programming methods. Research outcomes demonstrate the strengths of PSO in attaining optimal Pareto fronts with higher convergence rates and increased diversity of solutions, thus confirming its superiority in the management of water resource engineering projects over the currently employed EAs and statistical methods.

Establishing itself in multiple disciplines, such as humanities, engineering, chemistry, medicine, and physics, Particle Swarm Optimization (PSO) is gaining popularity as a population-based optimization algorithm [29]. Initially proposed in 1995, there have been numerous propagations of the PSO, along with many researches providing specific theoretical or empirical answers to its convergence and operationalizing questions with more than 500 variants. This study explores the effects of swarm sizes on the performance of eight variants of PSO without sticking to the common norm of less than fifty. Our approach remained consistent across 60 scalable benchmarks (10–100 dimensions) and 22 real-world (1–216 dimensions) problems, which allowed us to confirm that swarms with any size between 70 and 500 particles are effective in increasing the efficiency of PSO, especially for complex or multimodal problems. Unimodal problems can work well with a smaller swarm, though it has been demonstrated that other variants are more effective and better performed with larger swarms. This work argues that small swarm sizes, which are practically recommended by the body of literature, are not optimal for quickly tackling complex optimization problems.

Mixed-variable optimization problems (MVOPs) are definitionally those that involve both discrete and continuous choices in whatever context they are framed. The presence of such mixed variables also complicates the search space, therefore making MVOPs difficult problems to solve. The Particle Swarm Optimization (PSO) approach has been found to successfully tackle different types of optimization problems using a simple yet effective iterative approach. While there have perhaps been many approaches based on PSO, which deals with either continuous or discrete parameters in one framework, the focus has been on implementing the PSO approach on the prerequisites dealing with MVOPs lately. A new approach to MVOPs based on the PSO concept suggests the introduction of a mixing type variable geometry, which allows to interpret of the two types of parameters [30]. This approach utilizes different reproductive methods for continuous variables and discrete ones, employs the concept of adaptive parameter tuning, and utilizes a constraint-handling method to raise the efficiency of its operation. Experiments with 22 artificial and two real MVOPs showed that this PSO variant is the reliable and competitive approach to solving mixed-variable problems.

This research proposes a Dynamic-neighborhood-based Switching PSO algorithm (DNSPSO), which is characterized by an entirely new velocity update rule of personal velocity that uses a Distance-based dynamic neighborhood to vary the individual and global best positions [31]. This method makes effective use of the topological evolution knowledge within the population in order to assist the search. A switching learning strategy is also included in the model to adjust the acceleration coefficients and modify the velocity model according to the search status at every iteration in order to enhance overall search coverage. In addition, there are also components of the differential evolution algorithm that combine to reduce excessive premature convergence of the population. The effectiveness of the approach is illustrated on a set of multimodal, unimodal and rotated functions. The experimental results showed that DNSPSO performs significantly better than the original PSO algorithm in terms of accuracy and time while solving complex multimodal optimization problems.

In this study [32], the authors integrated a particle filter (PF) with a self-organizing particle swarm into a new PF-PSO approach, which they describe as a hybrid metaheuristic optimization algorithm. The algorithm is divided into two processes where: first, the particle population is set up randomly in the very beginning, and later, the search space is more focused on the needed accuracy in the solution. This algorithm is used to design proportional-integral-fuzzy controllers for the position control of electrical integral type servo systems, showing additional advantages such as reduced energy requirement for the fuzzy control system through the reduction of cost function values. The study concludes with a benchmarking of their algorithm against other metaheuristic algorithms on some standard test functions with the aim of exposing the strengths and weaknesses of the new PF-PSO algorithm not only on the canonical test cases but also on experiments.

The motto behind this optimization method is to put particles in place of the points and use the guiding function as the fitness function and the movements of particles as iterations. The result of the sequence of these iterations gives rise to an optimal or near-optimal solution for a specified problem. One major point of concern about PSO's application is that its algorithm is not suited for multi-objective functions. The contributions made in the paper are expected to help overcome the deficiencies in multi-objective optimization. The authors in [33] stress the need for plugins for AMPSO based on the study carried out on various benchmark functions in which AMPSO was the best-performing multi-objective PSO tester among a selection of five. The authors also talk about the offering competition mechanism, which is aimed at alleviating intrinsic competition among potential leaders who dominate and implement global learning.

This study surveys Particle Swarm Optimization (PSO) in stochastic inverse problems when working with geophysical data comprehensively [34]. The study brings together the most important characteristics of PSO and investigates major contributions across different overlapping fields of geophysics. It chronicles the progress of the PSO paradigm, which has relied on modeling of the earth beneath the surface, and gives reasons for their selection together with their merits and demerits. Different works from the geophysical literature cite the success of PSO in the processing of diverse electromagnetic (magneto telluric and time-domain), gravimetric, magnetic, self-potential, direct current, and seismic data. These case studies are critically compared to highlight the advantages and limitations of each of them as well as their unique aims and results. The paper also elaborates on how a multi-objective PSO can be applied for the coordinated optimization of various data through the incorporation of Pareto's optimality to eliminate potential inconsistencies.

Since LSOPs are expanded carious incremental scope and are rather likened to applicable life situations, they are drawing heavy attention. The vast search space and numerous local optima in LSOPs make it challenging to maintain both diversity and convergence for optimization algorithms. Owing to its high convergence rate, Particle Swarm Optimization (PSO) does well on certain LSOPs, though many an instance is marred by the tendency to get stuck in the local optimum. In order to resolve these difficulties, this research presents a reinforcement learning level-based particle swarm optimization, popularly known as (RLLPSO) to solve large-scale optimization problems. RLLPSO employs a level-based population structure that enhances population diversity as well as a reinforcement learning strategy that adjusts level numbers dynamically in order to improve the search. Furthermore, a level competition mechanism was incorporated to reinforce the convergence of RLLPSO. The conducted experiments on two large scale benchmark suites demonstrated that RLPSO outperformed five other algorithms in the large scale (LSOPs). The reported performances indicate RLLPSO to be suitable for devising solutions in LSOPs characterized by a high degree of complexity [35].

It's critical how the swarm-based algorithm modifies its search dynamics throughout its application in order to have a significant impact on solving real-world global optimization problems. There appears to be an urgency in searching unsearched areas of a problem domain in order to improve search efficiency, more so considering the wide popularity of PSO due to its great search potential. However, the algorithm, unlike many, tries to apply a balance between exploration and exploitation but sometimes gets stuck into bad local optimal solutions. In order to solve this weakness, a new algorithm, e-mPSOBSA, is created, which combines PSO and a modified Backtracking Search Algorithm [36]. This combination should enable one to use PSO at its strength in its search while using the strength of the modified BSA in geographic potential related to the types of searches that need to be carried out. The performance of the algorithm, together with that of 26 other advanced models, including PSO and BSA variants, was evaluated on the basis of standards benchmarks (the IEEE CEC 2014 and CEC 2017 benchmark suites). Convergence, diversity analyses, and statistical tests established that e-mPSOBSA performs better than other methods in global optimization. Further performance evaluation of the proposed algorithm evaluated its performance in terms of four unconstrained and seven constrained engineering design problems. e-mPSOBSA achieves superior performance on accuracy, search efficiency, and convergence rate, proving its effectiveness on complex optimization problems.

Flood forecasting is one measure that can either stop a flood from occurring or at least try to lessen the repercussions it will have. Different purposes have been developed regarding the cause and effect that rainfall and runoff have on each other. Most recently, artificial intelligence such as Artificial Neural Networks (ANN) has enabled the forecasting of rainfall-runoff models or hydrological time series, which is done on a long short-term memory (LSTM) network basis. However, it is often LSTMs that hyper parameters are set manually, and they don't always perform to their full potential. This work defines a PSO-LSTM model, where a particle swarm optimization greatly increases the learning of a data sequence by setting the hyper parameters of the LSTM correctly. The model was deployed to Jingle Watershed and Lushi Watershed to predict the chances of flooding occurrences by using rainfall and local runoff recordings [37]. For this purpose, the Nash-Sutcliffe efficiency coefficient, root mean square error, and bias were calculated, which proved the PSO-LSTM model to be superior to a vast majority of models, including the ones mentioned above. The PSO-LSTM model was particularly more successful in predicting occurrences 6 h and more away from the moment of forecasting and improving accuracy and stability as well. The conclusion drawn is that the PSO-LSTM model is a suitable candidate for short-term flood forecasting, as it performs consistently and with great accuracy.

PSO has achieved great success in solving various engineering problems; nevertheless, some issues, such as early convergence issues and inadequate global search ability are some of the limitations persistent with PSO. In order to cope with these issues, the multi-swarm Cooperation Particle Swarm Optimization, Specifically Evolutionary State Driven or simply ESD-PSO, is introduced [38]. This technique incorporates an adaptive multi-swarm cooperation mechanism (AMC) in order to strengthen communication among the local swarms. Under the AMC framework, the global population is split into several local swarms, each of which has the best particle used to evaluate the general condition of the swarm. When many sub-swarms haven't evolved a certain threshold, then the stagnant ones are adaptively clustered. Also, a stagnation compensation strategy (SCS) is introduced where a member of a cluster inert for a long period without adjustment stirs up a search diversity enhancing benchmark. A CDM (competitive disturbance mechanism) is also incorporated to enhance the solution's precision; similarly, The CEC2013 and CEC2017 test archives, along with three engineering optimization problems, were used to evaluate the adequacy of ESD-PSO. Results indicate that ESD-PSO is superior to the other PSO variants, which shows the good potential of the proposed model to overcome the drawbacks of the traditional PSO model.

Depending on the context, PSO is able to span multiple perspectives, and It is precisely this context that makes them arguably the most numerous in contrast to any other metaheuristic. This paper introduces a new PSO variant for continuous optimization known as the Generalized Particle Swarm Optimization (GEPSO) algorithm [39]. GEPSO broadens the scope of the original PSO by integrating two additional terms into the velocity updating rules that enhance particle interactions that are aimed at averting swarming within the vicinity and stimulating explorations of the unexplored areas of the search space. Also, there shall be formulated a variable particle inertia weight updating procedure in addition to the particles inters oil weight dependent procedures for fading control of swarm convergence. Bearing in mind the significant role of parameter settings for heuristic algorithms, the paper presents a detailed parameter tuning procedure for GEPSO. Well-posed benchmark functions were utilized to conduct the computational tests demonstrating that GEPSO is superior to the original PSO and its other variants, which include Repulsive PSO (REPSO), PSO with Passive Congregation (PSOPC), Negative PSO (NPSO), Deterministic PSO (DPSO) and LS-DF-PSO in terms of the average fitness value, standard deviations, and runtime factors.

In several industries, sparse and high-dimensional matrices are widely used such as latent factor analysis models. However, a straightforward learning rate setting for a single LFA model trained by stochastic gradient descent is often impractical, and therefore, the tuning must be made adaptive. They start by addressing this concern by inquiring into the stages of development of the particle swarm optimization (PSO) algorithm and then further augmenting it with more information so as to avoid the locally optimal solution from being reached too quickly within the same computational budget [40]. This initiative has resulted in the creation of the position-transitional particle swarm optimization ($P^2SO$) algorithm. In this study, a $P^2SO$-based LFA model, the PLFA model, has been developed, which generates a learning rate swarm of the same latent factors for FFA, allowing such learning rates to be adjusted on the fly such that high dimensional sparse matrices are well optimally reconstructed. Further, the PLFA model, unlike LFA models based on SGD approaches, does not require extensive re-learning of the parameters while offering higher accuracy of prediction for the missing data. Therefore, the PLFA model is more efficient in terms of prediction and computational cost in comparison with other adaptive LFA models, making it suitable for use in multifaceted industrial operations.

Cyber security systems are exposed to attacks and expectations of suspicious actions from one or more users or devices based on the status of the network or systems, and this, of course, requires the use of such systems between the network and every host device. This research proposes a new approach by incorporating grey wolf optimization (GWO) with PSO in aid of better feature selection in an Intrusion detection system (IDS) [41]. By incorporating PSO in the method to update the position of each grey wolf, the preservation of individual best positions was achieved, resulting in diminished risk of GWO being stuck in local optima. The technique was tested on the NSL KDD dataset, using K-means and SVM classifiers to assess accuracy, detection rate, false alarm rate (FAR), number of features, and execution time. The results indicate that both the K-means and SVM classifiers adopting the hybrid GWO-PSO approach realize significant benefits in improved IDS performance.

Notably, surrogate-assisted evolutionary algorithms (SAEAs) are suitable for optimizing problems that are expensive in terms of computation; however, most of the existing SAEAs are limited in terms of dimensionality, being applicable to the low or medium range at the most. This paper presents a surrogate-assisted multi-population particle swarm optimizer for high dimensional expensive optimization, SA-MPSO [42]. The SA-MPSO combines PSO with affinity propagation clustering to construct sub-swarms and an evolutionary mechanism utilizing individual or group best estimation. Particles that are promising from a model management strategy are moved for cost evaluations, the sub-swarm diversity enhancement, and some surrogate model-based search strategies promote effective random walks around the particles. SA-MPSO has been tested against benchmark problems of dimensionality 30 to 100 and airfoil design tasks and performed better than several methods of contemporary relevance.

Feature selection (FS) plays a crucial role in pattern recognition and data mining; however, traditional particle swarm optimization (PSO) algorithms face challenges when applied to high-dimensional FS problems. The reason for this is the poorly designed particle update and swarm initialization strategies. In this work, they introduce mutual information (MI) based feature selection that uses the bare bones PSO (BBPSO) algorithm [43]. The proposed algorithm proposes label correlation-driven swarm initialization techniques to the algorithm to enhance convergence speed by exploiting the association of features with class labels. To further improve the search strategy, two feature relevance-based local search operators, supplementary and deletion, are used. To assist particles in overcoming local optima, a wrap mutation operator is also applied. When tested on conventional datasets using the K-Nearest Neighbor (K-NN) classifier and compared with eleven state-of-the-art algorithms, this method not only yields an increase in the quality of chosen feature subsets but also improves the speed that makes it a viable FS method.

Feature selection has proven to be valuable in pattern recognition and data mining. However, common particle swarm optimization techniques using high-dimensional feature selection (FS) often encounter problems due to deficient particle update and swarm initialization mechanisms. This paper aims to design a novel feature selection algorithm that employs bare bones PSO technique with the help of INTs [44]. The paper implements a class label-based swarm initialization strategy, which seeks to accelerate the rate of convergence of the algorithm utilizing the relevance of the features with their respective class labels. Additionally, two local search operators, in this case, supplementation and deletion, which are based on feature relevance-redundancy, are used to enhance the exploitation of the search process. In addition, an adaptive flip mutation operator is proposed to enable particles to flutter around the optimal solution, reducing the chances of them being trapped in local optima. In comparison with ten other state-of-the-art algorithms when tested on the standard datasets, the feature subsets that were obtained using the proposed method in combination with the K-Nearest Neighbor classifier outperformed all the other ten algorithms, thus establishing this technique as one of the Competitive feature selection techniques.

Due to the resource allocation structure it has, effective scheduling of tasks in cloud computing environments, where there is a lot of competition in delivering the requested services, is the key to maintaining a high quality of service level. Scheduling is known to be NP-hard, that is, the problem of assigning the resources to jobs so that all jobs are completed in the shortest time possible. The paper presents structures for a method and an application of Improved Particle Swarm Optimization (IPSO), where task assignment is done autonomously and in an intelligent adaptive manner with the aim of improving particle swarm optimization's time use in cloud environments [45]. The proposed MALPSO introduces two groups of particles called ordinary and locally best particles to improve convergence and minimize diversity loss. Several performance measures, such as makespan, load balancing, stability, and efficiency, are used to evaluate the performance of the tested algorithm as well as on the CEC 2017 benchmark. The degree of shortening scheduling time and convergence to the optimal values for multiple criteria reached by MALPSO does make the differences in the already existing algorithms.

The Salp Swarm Algorithm (SSA) has been reported to have high performance in terms of speed when solving dimensionality complex optimization problems. However, it has been noted that it is not always well balanced in the exploration and exploitation of various functions. In this work [46], the authors proposed a combination (hybrid) method of SSA and particle swarm optimization (PSO), which is aimed at improving the search effectiveness while also attempting to resolve pitfalls that SSA inhales, such as premature convergence and low optima drunkenness. The hybrid solution improves the SSA by adding the velocity phase of the PSO, which increases the exploitation and the removal of the local minima. The proposed approach was validated on a wide range of standard functions and engineering problems, including, but not limited to, CEC 2005 and CEC 2017 benchmark suites, and it was found that the method was able to produce satisfactory results which are sometimes better than those produced in algorithms of other authors.

Liver cancer, specifically Hepatocellular carcinoma (HCC), is the most common liver cancer in Europe, Africa, and Asia. Early detection is crucial for the enhancement of the survival rate. This externally authored and peer-reviewed article proposes a hybrid machine learning model of HCC predicting based on the ensemble of Naive Bayes, Random Forest, Logistic Regression, and Linear Discriminant Analysis with a Deep Neural Network as a target model. Also, in this paper [47], the new algorithms, Dipper Throated optimization and Particle Swarm Optimization (DTPSO), have been introduced by combining the advantages of DTO and PSO to find the optimal solutions. The dataset is made up of 165 cases and consists of 50 features, after which data preprocessing such as preprocessing, oversampling, and normalization follows. The performance of the DTPSO-based model was evaluated using accuracy, MCC, F1 score, sensitivity, specificity, and AUC. The results further prove that the DTPSO-based model is efficient in predicting HCC early. Notably, the accuracy of models to achieve the DTPSO-based model was 98.52%, which is greater than that of the ensemble model at 96.00% and individual those of DTO, WOA, GWO, and PSO algorithms.

In order to overcome the drawbacks of slow convergence along with local optima in Particle Swarm Optimization (PSO), this paper proposes a large-scale bi-level PSO algorithm. Incorporating a multi-particle swarm into the initial population increases variety, which results in better search efficiency [48]. The two-level structure permits the upper swarms to make decisions and the lower swarms to optimize simultaneously, which increases the overall effective operation. The exponential acceleration factor has also been proposed to avoid early convergence in the last stage. From the results of simulations, it is proven that the bi-level PSO method is better in terms of performance and stability in comparison to the other standard PSO methods.

In high-dimensional areas where there are resource-intensive optimization tasks, solutions are required to be efficiently evaluated and rapidly found within a strict time frame. In many instances, surrogate models are incorporated into genetic optimization to optimize expensive problems; however, models usually do not scale well for high-dimensional cases. Therefore, this case study aimed to tackle this problem by presenting VSMPSO, which stands for Variable Surrogate Model-based Particle Swarm Optimization. This extends the maximum dimensionality of Particle Swarm Optimization (PSO) up to 200 dimensions with the use of a VSMPSOC binder. This enables the use of one surrogate model with VSMPSOC random sampling to navigate promising spaces between iterations and provides greater robustness through the use of a variable model management strategy that leverages the global model. Such mechanisms are generic to SAEA and applicable to VSMPSO on the whole. They are able to treat places with a high likelihood of being answered within a reasonable amount of time. Several benchmark problems, which range from 30–200 dimension space, suggested that VSMPSO was able to obtain computing exposure that was more effective and able to handle large amounts of data than many other cutting-edge models [49].

The effectiveness and straightforwardness of the implementation of Particle Swarm Optimization (PSO) leads to its widespread application in real-life problems. Nevertheless, even with the advantages, issues such as premature convergence and under-exploitation due to an uneven balance between exploration and exploitation are hard to neglect when dealing with complex optimization using PSO. This work addresses the issues discussed by blending adaptive strategy into PSO [50]. ASPSO has three characteristics such as (1) employment of a chaotic map and adaptive position updating mechanism for exploration and exploitation balance, (2) elite learning and dimensional learning strategies for population diversity enhancement, and (3) competitive substitution mechanism for improved accuracy of solutions. In tests conducted on CEC 2017 functions, ASPSO ranks higher than 16 other algorithms. Further, when faced with a real-world industrial melt spinning optimization problem, ASPSO again recorded better results than other methods in use today.

As the demands of modern expert systems increase, the effectiveness of automatic selection methods continues to be improved for systems that process image and signal data. Current binary PSO (BPSO) is lagging behind the continuous PSO technique since it confines particle locations in Hamming space, which decreases the divergence of swarms. However, it also causes the swarm to be trapped in a local optimum. They propose the EBPSO as a solution to the enhancement of PSO for feature selection, where they combine feature selection and position updating by varying position-switching probability and sufficient speed [51]. In statistics, the estimation of EBPSO1 and EBPSO2 is performed with regard to different calculations of the probabilities. Tested on eighteen benchmark datasets, EBPSO methods performed better in the classification accuracy compared to other feature selection methods, which shows that EBPSO is effective in terms of the selection of features that will result in enhanced classification accuracy.

Most existing investigations of the reliability–redundancy allocation problem (RRAP) target systems in a source-sink manner. A global reliability RRAP (gRRAP) is defined that extends the RRAP model and addresses imperfect nodes in this study [52]. Nodes and links are treated as subsystems in this model. The g-reliability measure, the probability of being able to connect all system nodes with functional links, is particularly appropriate for modern multi-source, multi-sink complex network systems. An algorithm is proposed to determine the parameters of the gRRAP objective function. In contrast, an improved particle swarm optimization algorithm like PIPSO with a particular particle initialization technique is constructed to optimize the gRRAP. This approach provides a metric for the measurement of distance between particles in RRAP. Experiment results show that PIPSO compares favorably with more than other popular gRRAP approaches, as well as previous studies on four benchmark problems. Interestingly, a new metric is presented here to evaluate the level of uniformity among particles' positions in the case of PSO application to RRAP, supporting the claim that a more uniform initial distribution of particles results in faster convergence and better stability.

There are certain drawbacks in adequately preserving indoor environmental quality (IEQ), whose parameters include temperature, humidity, $CO_2$, and VOC, while optimally using HVAC systems. Some works point out that energy is saved by the suitably efficient operation of HVAC systems. Still, very few of those explain those savings while taking the indoor environmental quality into account. This paper seeks to formulate and solve a complex optimization problem of reducing HVAC energy use in operation while ensuring that indoor air quality is not compromised. The method formulates a single optimization problem having multi-variable constraints, which are energy minimization and environmental quality, using the Particle Swarm Optimization technique. Accordingly, the algorithm is able to find a good balance between energy-efficient design and a high-quality indoor environment. For this purpose, seasonal changes in construction and target indoor conditions are used [53]. The contributions showed a potential 7.8% reduced energy consumption on indoor settings adjusted to ASHRAE standard 55 and AS 1668.2 on indoor temperature limits, level of humidity, and $CO_2$ and VOC concentration at the same time.

In data classification, complexities arise when a large number of attributes are recorded as some would be irrelevant or redundant, thus lowering model efficiency. To solve this problem, feature reduction is used which cuts off any irrelevant and noisy attributes in the dataset while selecting the essential ones to improve accuracy. This study applies Particle Swarm Optimization (PSO) for feature reduction on a cervical cancer dataset. The results of the experiments show that through PSO utilization, irrelevant features are eliminated, and the feature count is therefore reduced from 36 to 17, which increases the classification accuracy [54].

The ability and capacity of enterprises to compete with their rivals is often dependent on efficient discriminatory practices in regard to their customers. In this context [55], the present study proposes a method for customer segmentation that is based on K-means clustering and its particular improvement with an algorithm which is called ALPSO. The concern about the tendency of standard PSO to get trapped in local optima is overcome by ALPSO which incorporates adaptive learning factors as well as modified inertia weight and a restructured position update method. The proposed ALPSO algorithm also solves the concern of clustering analysis, which is dependent on the user selection of initial centers, by integrating it with the K-means algorithm (KM-ALPSO). The study employs a purchasing pattern of consumers that is based on a partial grape data set. This method also combines numerical variable extraction by means of factor analysis and similarity measurement for clustering. The performance of the original five datasets from UCI benchmarks, as well as the final version of improved normal K-means clustering, which is KMA-ALPSO (IKM-ALPSO), is much better than other models for customer segmentation processes and applications.

This paper first turns to fuzzy modeling of feature selection, with specific reference to fuzzy cost objective feature selection using particle swarm optimization (PSOMOFS) [56]. It incorporates a fuzzy dominance relation to evaluate candidate particles and a fuzzy crowding distance metric to control the elitist archive and to determine the global leader. A tolerance coefficient is included that aims at ensuring that Pareto-optimal solutions are acceptable in relation to what (the) decision-makers require. Using a number of datasets from the UCI repository, the PSOMOFS has been evaluated using three fuzzy multiobjective evolutionary methods and compared with three standard multi-objective feature selection methods. Results indicate that PSOMOFS creates better-performing approximated, diverse, and cost-effective feature sets.

In this study [57], they present a new type of PSO algorithm that has been modified with the inclusion of an adaptive weighted delay velocity (PSO-AWDV). The strategy is to consider a new weighted delay velocity scheme in a new PSO algorithm (PSO-WDV), which also incorporates a means of adaptively modifying the velocity inertia weight for the swarm velocity modulation depending on the swarm state. On benchmark test functions, it was found that PSO-AWDV outperformed other intelligent optimization algorithms and various established forms of the PSO, which proved its correctness.

Networks need to have mechanisms for detecting different kinds of intrusions focused on IoT, such as those orchestrated by attacking or zombie hosts. In this paper [58], a framework referred to as APSO-CNN is presented. In this case, the one-dimensional CNN is optimized by PSO where inertia weight is adjusted according to a formula whereby the fitness value is essentially the cross-entropy loss computed from the validation set after initial training. The proposed method is a new evaluation method for APSO-CNN, which is evaluated with manually parameterized CNN (RCNN) and three additional metrics, parameters, and algorithms using conventional methods and accuracy over ten separate trials. Simulation results validate the efficacy and reliability of the proposed model in detecting a wide range of attacks within IoT networks.

This paper presents a new multi-objective PSO method with the application of the PSO concept to feature selection. In this approach [59], feature vectors are represented as particles that move within a two-dimensional optimization space. Particles are further classified into dominated and non-dominated bands, and particle ranks, together with feature ranks, determine the updating of velocity and position in each optimization iteration. The method was implemented in 16 datasets and compared with 11 other methods that claim to be state-of-the-art in feature selection and multi-objective optimization. The experiments show that the proposed algorithm is able to locate optimal Pareto fronts that are very close to the initial point of the optimization space. Evaluation metrics confirm promotion, obtaining the maximum Success Counting Measure on 13 datasets, the biggest value of C-Metric on 14 datasets, and 13 datasets highest Hyper–Volume Indicator. Results of the statistical procedure based on the Mann–Whitney U-test also indicate that the method is effective, as it has a better performance than its counterparts in 38 pairs in 55 pairwise tests.

As the interest in distributed generation increases, the placement and the size of the generators become crucial determining factors of the efficiency of the network. This work presents an integrated genetic algorithm and particle swarm optimization (PSO) technique for the optimal placement of distributed generators in order to minimize network losses or improve the voltage profile of the network [60]. A hybrid genetic-particle swarm optimization algorithm (GA-PSO), which combines genetic algorithm and PSO in one general population, is developed to harness the benefits of both techniques. The method was applied to the IEEE 33 and 69 bus systems, employing a system that computed weight coefficients of all the objectives ascribed to each goal so that the intervention of the human factors was removed during optimization. The comparative results reveal that the hybrid GA-PSO performed remarkably better than the previous methods. In terms of iteration numbers and standard deviations, all the analyzed cases show improvements.

Effective management of reservoirs and proper planning for irrigation necessitate accurate four estimations. They propose a new model called ELMAPSO-GWO, which combines psogp, wolf-pack, and extreme learning for the prediction of runoff, which is outflow to the Mangla watershed, Pakistan. The Mode's performance has been tested against ELM, ELM-PSO, and ELM-GWO and binary hybrid (Gravitational Search Algorithm) Models [61]. The authors explain how, through a time series of Mosprecip and Mosrunoff, ELM-PSOGWO was able to reduce RMSE by 38.2%, 22.8%, 22.4%, and 16.7% vis-a-vis ELM and ELM-PSO, ELM-GWO, ELM-PSOGSA respectively. More importantly, the models based on ELM imbeds PSO and GWO have also enabled rooted mean squared error levels to be considerably reduced from 19.9% up to 20.3% when the ELM is used alone. Further enhancing the models' accuracy, more precipitation data was assimilated, with ELM PSOGWO again attaining superior performance in peak runoff prediction with the lowest average relative error. From these results, it can be concluded that ELM-PSOGWO is a suitable model for predicting monthly runoff.

Feature selection is an important problem in data mining and machine learning, however, privacy issues can complicate the problem of selecting high-quality subsets of features. To bridge this gap, this paper proposes a framework of federated feature selection built around federated learning (FL) principles, in which a neutral agent collects selected features from different participants but does it in a privacy-preserving way [62]. Within this framework, they develop a federated evolutionary feature selection approach based on Particle Swarm Optimization (PSO) to address feature selection problems in a multi-participant environment under privacy constraints. In particular, the two privacy-preserving strategies incorporated in the proposed framework—a multi-participant feature assembling strategy and a swarm initialization strategy that is based on constructed solutions—are effective. Experiments undertaken on 15 datasets conclude that the performance of the proposed method leads to a further enhancement of the classification accuracy while all the participating individuals' data stay confidential.

The improvement of energy performance in the building sector is crucial for the decrease of consumption of energy in the highly energy-intensive sectors. A new multi-objective evolutionary technique is developed in this study where two objectives have to be optimized simultaneously: energy consumption and comfort of the users of the building. An entirely new solver is proposed that is based on the principles of bare-bones particle swarm optimization (PSO), because it does not require enough complex control mechanisms. The drawback of the local convergence of traditional algorithms in PSO is solved by a self-adaptive perturbation updating strategy. By collapsing control parameters such as inertia weight and acceleration coefficients, the algorithm is used [63]. Experimental results indicate that this algorithm is much more effective in finding energy-saving strategies that are not dominated by other factors as compared to competitive ones. The results, when applied to several buildings in China, show that the new algorithm yields better results than comparable methods that have been used in previous investigations. Most importantly, the algorithm reduces uncomfortable hours to about 11.82% and allows the use of an extra 1.74% energy consumption, showing that the algorithm can be used positively for building energy optimization strategies.

Projections about groundwater levels have to be made with a high sense of accuracy to ensure proper cross-use of the available water resources. In this study, a simulation-optimization hybrid model is constructed, which incorporates integrated particle swarm optimization and support vector regression (SVR-PSO) to predict the levels of groundwater [64]. The model was applied to the Zanjan aquifer system in Iran and was found to be superior to the Bayesian and standalone SVR models. Two approaches were used: one involving hydrograph time series extraction via the Thiessen method and another predicting time series data for each of the 35 observation wells individually. Results indicated that the SVR-PSO hybrid model outperformed other models while achieving lower RMSE values and higher determination coefficients. More specifically, in the first approach, the model provided the prediction of the monthly groundwater levels with a gross MSE training value equals to 0.118 m and a gross MSE testing value equals to 0.221 m. In the second approach, the reduction of RMSE was achieved in 88.57% of the wells as compared with the rest of the models tested in this study, suggesting the potential of the SVR-PSO hybrid model in aquifer management decision support systems.

Personal credit scoring is a multifaceted area where the latest credit scoring models, indicating a strong trend of employing machine learning methods in finance, show significant promise. To explain the problems arising from tuning the XGBoost hyperparameters, this paper proposes an adaptive Particle Swarm Optimization (PSO) based XGBoost credit scoring model [65]. To achieve better sub-swarm divergence and prevent convergence to local optimum, Adaptive PSO has incorporated strategies involving cluster-based swarm division and dual learning strategies. The model was applied to a total of four credit datasets and seven KEEL benchmarks; how it compared standings against traditional gambling machine learning algorithms, other popular ensemble classifiers, and even hyper-parameter tuning methods such as grid search, random search, tree-structured Parzen estimator, and PSO. The proposed model's evaluation metrics, including accuracy, error rates, Brier score, and F1 score, indicate a much better comparative performance of the model, quite particularly across datasets with adaptive PSO performing relatively better than the other TPOT approaches employed across the datasets.

This paper highlights the benefits of adding embedded energy systems via tri-generation, even though the design of the integration matrix is apparently problematic [66]. The proposed approach details a systematic process for constructing the coupling matrix while forming the Demand Management System (DMS) for an Energy Hub. To offset non-linear factors such as stack constraints, a novel PSO is designed for optimal energy systems aggregation. The approach was applied to a residential area with integrated cooling heating, and electricity and validated for modeling and computation. Moreover, the paper discusses the effects of the demand response and energy storage systems on energy management strategies. An overview of the given studies is presented in Table 1.

**Table 1:** An overview of the given studies

| Study focus | Algorithm/Model | Application | Key findings | Reference |
|---|---|---|---|---|
| Optimization algorithms comparison | PSO *vs.* DE variants | 25 artificial & real-world problems | DE often outperforms PSO, challenging the literature bias favoring PSO. | [27] |
| Water resource optimization | 22 PSO variants | Reservoirs, runoff & groundwater modeling | PSO excels in water resource projects with optimal Pareto fronts and high convergence. | [28] |
| Swarm size impact on PSO | PSO with varied swarm sizes (3–1000) | 60 benchmarks, 22 real-world problems | Larger swarms (70–500) improve efficiency, especially in multimodal problems. | [29] |
| Mixed-variable optimization | PSO with mixed-variable encoding | MVOPs | PSO effectively handles both continuous and discrete variables in MVOPs. | [30] |
| Dynamic-neighborhood PSO (DNSPSO) | DNSPSO with switching learning strategy | Multimodal optimization | DNSPSO surpasses standard PSO in accuracy and speed for complex problems. | [31] |
| Hybrid PSO-PF algorithm | PF-PSO hybrid | Fuzzy control for servo systems | PF-PSO reduces energy use and outperforms standard metaheuristic algorithms. | [32] |
| Multi-objective PSO | Adaptive Multi-objective PSO (AMPSO) | General optimization | AMPSO performs best among tested algorithms for multi-objective functions. | [33] |
| Geophysical data modeling | PSO with multi-objective optimization | Geophysics | Multi-objective PSO enables optimization across multiple geophysical datasets. | [34] |
| Large-scale optimization | RLLPSO with reinforcement learning | LSOPs | RLLPSO enhances diversity and convergence in large-scale problems. | [35] |

(Continued)

**Table 1 (continued)**

| Study focus | Algorithm/Model | Application | Key findings | Reference |
|---|---|---|---|---|
| Hybrid PSO-BSA algorithm | e-mPSOBSA | Global optimization | e-mPSOBSA outperforms 26 advanced algorithms in search efficiency and accuracy. | [36] |
| Flood forecasting | PSO-LSTM hybrid | Flood prediction (Jingle & Lushi watersheds) | PSO-LSTM outperforms existing models, especially for long-term predictions. | [37] |
| Adaptive multi-swarm PSO | ESD-PSO with multi-swarm cooperation | CEC benchmarks & engineering problems | ESD-PSO improves convergence and stability over standard PSO variants. | [38] |
| Generalized PSO (GEPSO) | GEPSO with two new terms for interaction | Continuous optimization | GEPSO surpasses standard PSO variants on various benchmark functions. | [39] |
| Sparse matrix reconstruction | $P^2SO$ with adaptive learning rate | High-dimensional datasets | $P^2SO$ enhances prediction accuracy without requiring intensive parameter tuning. | [40] |
| Intrusion detection | GWO-PSO hybrid | Network intrusion detection | GWO-PSO hybrid shows improved IDS performance on NSL KDD dataset. | [41] |
| Surrogate-assisted PSO (SA-MPSO) | SA-MPSO for high-dimensional problems | Expensive optimization tasks (30–100D) | SA-MPSO demonstrates high-quality solutions for high-dimensional problems. | [42] |
| Feature selection with MI & BBPSO | BBPSO with mutual information | Data mining | BBPSO improves speed and quality of feature selection in high-dimensional data. | [43] |
| Feature selection & Hamming space | Enhanced BPSO (EBPSO) | Pattern recognition | EBPSO achieves better classification accuracy on benchmark datasets. | [44] |
| Resource scheduling in cloud computing | MALPSO for task scheduling | Cloud computing | MALPSO shortens scheduling time and balances load efficiently. | [45] |
| Hybrid SSA-PSO algorithm | SSA with PSO's velocity phase | Engineering problems & benchmarks | SSA-PSO improves on SSA by avoiding local minima and enhancing exploitation. | [46] |
| Hepatocellular carcinoma prediction | DTPSO with ensemble models | Medical (HCC prediction) | DTPSO-based model achieves 98.52% accuracy, outperforming other models. | [47] |
| Bi-level PSO algorithm | Large-scale bi-level PSO | LSOPs | Bi-level PSO shows better stability and performance than traditional PSO methods. | [48] |

(Continued)

**Table 1 (continued)**

| Study focus | Algorithm/Model | Application | Key findings | Reference |
|---|---|---|---|---|
| High-dimensional surrogate model PSO | VSMPSO | 30–200D problems & airfoil design | VSMPSO outperforms traditional SAEAs in computational efficiency. | [49] |
| Adaptive strategy PSO (ASPSO) | ASPSO with elite & dimensional learning | Industrial optimization | ASPSO shows superior performance in complex, real-world optimization. | [50] |
| Binary PSO for feature selection | EBPSO for Hamming space optimization | Data mining & classification | EBPSO surpasses existing FS techniques in accuracy across datasets. | [51] |
| Reliability-redundancy allocation problem | PIPSO with specific particle initialization | Network reliability optimization | PIPSO shows faster convergence and stability than standard RRAP methods. | [52] |
| HVAC energy optimization | PSO with indoor quality controls | Building energy management | PSO optimizes HVAC energy use, reducing total energy consumption by 7.8%. | [53] |
| Feature reduction in classification | PSO for feature reduction | Cervical cancer dataset | PSO reduces irrelevant features, improving classification accuracy. | [54] |
| Customer segmentation | ALPSO with K-means | Market segmentation | ALPSO outperforms standard models in clustering accuracy and efficiency. | [55] |
| Federated feature selection | PSOMOFS | Privacy-preserving FS | PSOMOFS enhances FS accuracy while maintaining data privacy. | [56] |
| Multi-objective building energy optimization | BBPSO with adaptive perturbation | Energy optimization | BBPSO reduces uncomfortable hours by 11.82% with minimal energy increase. | [57] |
| Groundwater level forecasting | SVR-PSO hybrid | Aquifer prediction | SVR-PSO outperforms Bayesian and SVR models in accuracy. | [58] |
| Credit scoring with XGBoost | XGBoost with adaptive PSO | Financial credit scoring | Adaptive PSO-XGBoost surpasses other models in accuracy and F1 score. | [59] |
| Integrated energy system planning | Energy hub model with PSO | Urban energy supply | Proposed PSO improves energy planning, considering demand response and storage. | [60] |

### *2.3 Limitations and Challenges in Traditional PSO*

Despite the simplicity, adaptability, and effectiveness in solving many optimization problems that Particle Swarm Optimization (PSO) possesses, it also has its weaknesses the challenges. These weaknesses can influence PSO's effectiveness in terms of the quality of solutions obtained, considering it is feeding on complex high dimensional data space or constrained optimization problems. Formulating an understanding of these weaknesses would be useful, especially for researchers and practitioners, who normally tend to seek solutions by making variations and improvements to the system standard PSO.

a) Premature Convergence

PSO is faced with several problems, but one of the largest ones is known to be premature convergence [51]. Premature convergence happens when the animals in the swarm need to take more time to fully scan the search space and settle for the first local optimum they find, which leads them not to be able to find the true best solution. This situation is especially nasty in the case of multi-modal optimization problems because they tend to have more than one optimal point, making it harder to avoid a scenario of premature convergence. The following are some of the contributors to the problem of premature convergence in PSO strategies:

- High Social Influence: The symbolism of the social part of the PSO is the global best position to which individual particles are pulled or which they try to reach. When the social component overcomes the cognitive component, people inside the swarm tend to displace instantly fast, making significant alterations to their position. When particles begin to have an excessive amount of dependence on the global best particle, dissection about the global best position occurs, and later on, they are forced to gather around this position.
- Insufficient Exploration: PSO may be deficient in exploration, especially when the particles' velocity wanes as they approach the solution. In such a case, the particle's space is overly explored, and the population may cluster in multiple local optima.

b) Sensitivity to Parameter Settings

PSO is largely dependent on the settings of its parameters, which include weight, inertia, *c1* cognitive coefficient, and *c2* social coefficient. These factors determinatively affect the equilibrium between exploration and exploitation, which is significant in reaching a desired outcome. Not spaced parameters may produce poor results or the procedure may not finish. For example:

- High Inertia Weight: High centralization induces greater levels of active inertia. The only restriction is that this may be set too high, resulting in overshooting of all the optimal solutions.
- High Cognitive or Social Coefficients: This issue can lead to excessive social excess or cognitive overemphasis, highlighting national characteristics and resulting in conflict and flight.
- Low Parameter Values: Flying too low on parameter values may negate the experience, leading to a greater degree of lower weight.

Different settings of optimal parameters can be adjusted for various problems, but locating such values may take a lot of time and could involve trial and error or empirical adjustment.

c) Trouble with the High-Dimensional Search Space

However, in the high-dimensional context, the search space expands exponentially, which makes the convergence of the solution rather troublesome for PSO [67]. Basic PSO has some limitations in high dimensions as well:

- Curse of Dimensionality: As the particle number increases in dimension, so does the search region, making it difficult to search and find optima efficiently. Instead of finding the optima, the particles waste too much time trying to get to the unimodal areas, reducing the efficacy of PSO.
- Reduced Convergence Rate: In larger-dimensional settings, the cognitive and even the social components may not be strong enough to move particles to the sought-after areas, leading to slow movement.

d) Inability to Adapt to Dynamic and Noisy Cases

Classical algorithms were developed to try and solve non-changing problems where the function being targeted is absolute throughout time. At the same time, many real-life applications encounter noisy and unstable problems. In noisy scenarios, the best goal might be available sometime later, meaning the system must be flexible. The classical algorithms do not implement any functional control that would take this change into account causing the solution to be irrelevant for the new circumstances.

In addition, since noise is brought about by random perturbation in the environment, it also leads to problems. Noise may also be a particle's faulty guide in navigating the fitness realms, and for this reason, convergence is inconsistent, or global optima is not reached.

e) Insufficient Attention to the Constraints

Many optimization problems are subject to a set of equational or geometrical restrictions, which must be satisfied for the solution to be considered feasible. Standard PSO has, indeed, no native mechanism for dealing with constraints, which limits its use on nonlinear programming problems. In PSO:

- Boundary Violations: After a few iterations, particles may tend to escape to the unfeasible range. Thus, additional constraints are necessary to enforce the feasible region.
- Constraint Handling Techniques: Different methodologies, such as penalty methods, which direct particles at the region of interest, may be required, but these are generally problem-dependent and make the algorithm overly complicated.

f) The Efficiency and the Convergence Rate

The speed with which particles move toward the initial centroid of a distribution is quite remarkable [68]. Still, subsequently, this speed drops in the last stages of the search in most cases, especially in grand landscapes. Some such aspects include:

- Diminished Step Sizes: The closer a particle is to the best position, the greater the loss of velocity for the ego-pulling particles, and the movement slows. This, at times, renders the late search stage quite ineffective.
- Exploration-Exploitation Balance: This global-exploitation balance needs to be strengthened locally due to convergence's reliance on fewer iterations in most cases. Traditional Particle Swarm Optimization (PSO) does not have these dynamically adjustable mechanisms, which tend to hinder the convergence time or lead to a less-than-optimal solution.

g) Tendency to Require Hybridization for Improved Performance

In light of the challenges pointed out above, it is often the case that traditional PSO is wedded with modification by the use of other optimization methods. Some common hybridization strategies include the following:

- Incorporation of Local Search Methods: The use of PSO alone is often inadequate. Hence, using local searching techniques such as Simulated Annealing can enhance particles' exploitation features and prevent them from being trapped in local optima.
- Adaptive Mechanisms: When PSO is adjusted to various problem types, hybrid systems with better performance have been noticed. These systems use adaptive methods and PSO with variable inertia weight and convert coefficient during the search process.

Acknowledging these constraints, investigators have sequentially developed many variants and modifications of PSO to overcome the mentioned problems, which further extends the applicability range of PSA to more complex, dynamic, and constrained optimization problems.

## 3 Comparative Analysis of Optimization Algorithms

In this section, one will find the comparison of Particle Swarm Optimization (PSO) and other optimization techniques. Given the wide range of optimization problems, a proper analysis of PSO should be carried together with other algorithms to understand its advantages and drawbacks in efficient solving of different problems. This comparative analysis is structured into three main areas: the comparison area's selection criteria, an overview of alternative algorithms, and the metrics used to evaluate the performance of the algorithms.

### *3.1 Criteria for Comparison of Approaches*

Some criteria have to be taken into account to ensure that the comparison is done in a manner that will be useful in the context of a set of practices. The selection criteria are as follows:

Algorithm Characteristics: The algorithms chosen for comparison will have to incorporate some features of all classes, including the population of the set and a single-member optimization technique. Since PSO is a population-based algorithm, it was compared with evolutionary algorithms that have different exploration and exploitation mechanisms, genetic algorithms, SAODE, and ant colony in particular. This diversity provides a balanced view regarding solution-finding strategies.

Problem Complexity and Type: To make a strong comparison, we provide a set of problems for the algorithms to be assessed, selecting from continuous, discrete, multimodal, and high-dimensional problems. PSO is particularly known for its ability to operate in continuous optimization, whereas some algorithms, such as ACO, perform discrete problems. Therefore, it is worth investigating how each one can cope with different types of problem landscapes.

Computational Efficiency: Real-world problems typically contain limitations such as time or resources, and therefore, the practical efficiency of each algorithm pair, such as time, memory, and scaling, should be important in the comparison. This criterion helps in deciding which algorithms will be more appropriate for less resourceful applications.

Parameter Sensitivity: For instance, performance optimization algorithms are rather complex and require to be adjusted to work correctly. Thus, we consider strategies that are parameter-dependent and strategies that are not. For instance, parameters such as inertia weight govern PSO's efficacy, while other algorithms, such as DE, are not heavily influenced by parameters. This component enables us to comprehend how user-friendly each algorithm in practical application.

Adaptability to Dynamic and Noisy Environments: Optimization problems with changing objectives or noise in the fitness landscape are also common. To ascertain how robust particular algorithms are, we select the algorithms developed for use in a dynamic environment, as PSO tends to be challenged in this regard. In contrast, other algorithms might be more capable.

Dealing with Constraints: Several optimization problems will include constraints, and the ways constraints can be managed vary across algorithms. For example, in the case of PSO, some form of additional mechanisms is needed for constraint handling, while other algorithms could be more suited to handling constraints by design.

### *3.2 Other Optimization Algorithms Considered in the Study*

In this study, we compare PSO with several other optimization techniques that are almost universally used in different fields. We focus on some aspects of the algorithms' principles and mechanics to better understand the comparisons that follow.

a) Genetic Algorithms (GA)
   - Principles: Genetic Algorithm is one of the evolutionary algorithms that are based on the Darwinian law of survival of the fittest [69]. It employs a model of biological evolution that employs selection, crossover, and mutation operators. A genetic algorithm maintains the population of solutions and evolves that population through a number of generations.
   - Strengths: GA is regarded as robust and, more importantly, efficient in combinations and discrete problems. It is versatile in finding the global optimum in even the most difficult, multi-modal topographies.
   - Limitations: It is costly in terms of computations, more so in high dimensional space, GAs may take much time to converge. It requires parameter tuning for the right proportions of crossover and mutation rates to avoid over-exploration or over-exploitation.

b) Ant Colony Optimization (ACO)
   - Principles: Ants deposit pheromones on the paths they follow, determined by the pheromones laid by other ants, which serve as indications or memories of previously traveled pathways, which in turn affects the pathways taken by other ants [70]. This behavior aids ants in locating the most direct routes, which is the strategy implemented in any way to aid in the process of optimization problems in particular cases involving discrete search spaces.
   - Strengths: The ACO is highly effective, of course, in solving travel problems" such as The Traveling Salesman Problems within reason. It has a fairly high success rate in solving graph-based pathfinding problems whenever certain limitations are placed on ACO.
   - Limitations: Lacking in some cases, for instance, the vastness of the search space required by the ACO can lead to an early convergence problem, which in this context is a drawback. The ACO is sensitive to various issues, such as the rate of pheromone evaporation, and it tends to be expensive when used in number-rich tasks in a continuous looping manner.

c) Differential Evolution (DE)
   - Principles: Wider ways of solving optimization problems through the application of population-based techniques seem to be solving the issues of solutions when evolutionary intermediation techniques are incorporated. In comparing and combining solutions through the scaled differences of the effective population solutions, candidates get generated and retained for use.
   - Strengths: With simple implementation, DE is good enough for generic optimization and space occupancy issues. It is said that DE has a higher rate of convergence and can tackle high-dimension requirements with fewer parameters compared to other models.
   - Limitations: The DE algorithm has been observed to have limitations regarding optimization tasks that are discrete and especially sensitive to input parameters. It was also noted that crossover and scaling factors should be suitably adjusted to derive optimum results.

d) Simulated Annealing (SA)
   - Principles: Simulated annealing (SA) is described as a probabilistic optimization algorithm and a single-solution-based process informs that of annealing in metallurgy. The annealing schedule of SA would accept new solutions irrespective of whether they were an improvement over the best current solution or not, as long as this was considered beneficial to search for a larger range of options.
   - Strengths: This annealing strategy further makes SA robust since acceptable near-optimal solutions can be located over a wide range over the multimodal. It is also helpful for combinatorial and other discrete problems.
   - Limitations: Since SA is only focused on one candidate solution, it may take a decisive period to get to the optimum possible solution, a slow converging speed. SA's dependence on the annealing schedule: adjustment is needed to allow for better exploration and convergence.

e) Whale Optimization Algorithm (WOA)

- Principles: Skills Learned through Hunting Techniques (Bubble-net) of humpback whales meet the WOA [71], which focuses on surrounding the target, mechanism of spiral shrinking, and updating positions. It is quite new, but various continuous optimization tasks have found WOA to be promising.
- Strengths: WOA, on the other hand, has been found to be adaptable, and despite the complexity of certain search spaces, WOA exhibits good convergence properties. Multimodal problems also seem to be within reach for WOA, and solutions are competitive.
- Limitations: To be successful across problem types, WOA might require hybridization or adaptive strategies. It is also sensitive to parameter tuning, especially for high-dimension cases.

### *3.3 Evaluation Metrics for Performance Comparison*

For the performance evaluation of PSO and the chosen algorithms, several normalizing metrics are considered since they encompass the overall view of the usefulness, efficiency, and adjustability of these algorithms. Accurate determination of such metrics helps in establishing the relevance of the algorithms for a given optimization task.

a) Accuracy: Evaluate the imbalance or discrepancy between the optimized result to the expected or true value. The importance of accuracy has to do with evaluating the effectiveness of every algorithm independently since there are tasks that require a high level of accuracy, such as optimizing engineering design and tuning machine learning model parameters.

b) Convergence Rate: The speed with which an algorithm gets to the desired result or gets closer to it within a specified period or a preset number of repetitions. Faster Convergence of Algorithms means less time is required to arrive at the optimum or nearly the optimum, which is advantageous, especially in cases where time is an important factor. PSO tends to converge at early stages easily, while DE methods are very fast in convergence in continuous optimization.

c) Computation Time: The elapsed duration of the system from initiation of the algorithm to the moment it has been able to obtain the required optimum or acceptable solution. Computation time measures an algorithm's efficiency. For any practical use that has limits, such as computational power, algorithms that deliver good results while consuming shorter periods are desirable. This metric also makes it feasible to compare a particular output with other algorithms in terms of efficiency and resource optimization.

d) Exploration *vs.* Exploitation Balance: Gives broadness and depth of the algorithm in regard to solutions, i.e., how far and wide all areas (exploration) on great solutions are being combined or averaged (exploitation). It is rather clear that creating an optimal balance between exploration and exploitation lies at the heart of the success of seeking optimal solutions over complex landscapes. There are Sweet spots between PSO and GA due to their balance, while SA may be accurate at times but tends to use much exploitation due to its single solution search-based methods.

e) Robustness and Adaptability: Robustness expresses algorithmic consistency and performance across problem variations of types and levels of noise, while adaptability denotes the performance of a problem in a shifting or dynamic environment. It is worth noting that algorithms that are robust and work effectively in different environments and conditions easily adapt to dynamic environments. The deficiency of PSO to adapt would be a disadvantage while executing dynamic tasks; however, GA and DE, the majority of the time, are rather robust due to varying problem types.

f) Scalability: The capacity of the algorithm to function as the size or dimensionality of the problem increases. The search space will increase for high-dimensional problems, and thus, some algorithms may perform poorly when the problem complexity changes. The two algorithms, DE and PSO, are said to have high scalability and, therefore, can be applied to high-dimensional problems, whereas ACO may not perform well in such cases.

These criteria enhance the comparative analysis by extending the understanding of each algorithm's computational performance across the different optimization problems tackled by the authors.

## 4 Application Domains

Given its flexibility, PSO has been widely applied in a variety of fields and has proven capable of exploring complicated search spaces. This section identifies the most applicable areas of PSO application, comparing its performance with other algorithms and analyzing the problems covered in each area at the domain level. Detailed sections on each application domain, each with a table of results comparing PSO to other algorithms:

### *4.1 Data Mining: Feature Selection*

Feature selection is much more important in data mining and machine learning, especially in high-dimensional data sets [72]. In feature selection, a number of the input variables or features are reduced in size by choosing a smaller portion that maintains or increases the model's performance. This not only helps simplify the model, thus making it faster and more interpretable, but it also helps reduce overfitting by removing irrelevant features or redundant features.

a) Importance of Feature Selection in Data Mining

Learning in high-dimensional spaces is not uncommon, particularly in text, image, or bioinformatics. Feature selection is used to solve the following problems:

- Reducing Dimensionality: Feature selection decreases the number of features, thus increasing the storage efficiency and enhancing computation performance.
- Improving Model Accuracy: The more informative features are considered, the more focused the model becomes, which in most cases leads to better accuracy.
- Increasing Interpretability: Reducing features simplifies the process of making assumptions about the model, which is more applicable in the medical and financial fields.

Feature selection can be divided into three main sub-categories: Filter Methods: This consists of selecting features that fulfill a threshold that is based on any statistical criterion (correlation, mutual information, etc). Wrapper Methods: Obtain evaluation from a predictive model by testing and identifying feature subsets and selecting the best performer or performers. Embedded Methods: Consider feature subset selection as other models incorporate, such as regularization in linear models whereby the model automatically recognizes and employs the optimal subset.

b) Particle Swarm Optimization (PSO) for Feature Selection

It is certainly not news to the community of analytics and data science that PSO has emerged as one of the most powerful feature selection methods. PSO searches for suitable feature subsets and classifies them into standard PSO particles during the optimization. But moving on from the general to the specifics, let's define how many PSO parameters there are in terms of particle sensing or feature selection models. For example:

How is a particle $p\_i$ defined: The vector value of particle $I$, i.e., the position vector, $x\_i = [x\_i1, x\_i2, \ldots, x\_id]$, is a vector of feature inclusion, a defined subset of features. $A$ set of such vectors contains individual elements $x\_ij$, which can be defined in binary or even contained in a scaled number (0 or 1), which represent the absence or presence of feature $¢\, j$ within feature selection, respectively.

How is the fitness function defined? The fitness function is used to assess or judge the performance of the determined classification accuracy level and size of the feature subset). One of the popular definitions is:

$$Fitness = \alpha^* \left(1 - Accuracy\right) + \beta^* \left(Number\, of\, Selected\, Features / Total\, Features\right) \tag{3}$$

where Accuracy: The classification accuracy of a model (e.g., a k-NN classifier or a decision tree), utilizing the features selected by the model. Number of Selected Features: Total number of features from the pool that the particle has selected. α and β: Coefficients that are used to inform the relative significance of accuracies versus the reduction of features. The emphasis in the model is that of a high alpha in accuracy in comparison to beta, which favors low dimensionality.

c) Equations in PSO for Feature Selection

Velocity Update Equation: Analogue to a drifting ship in the ocean, the movement of each particle in the PSO is also dependent upon its inherent characteristics, which is represented by its velocity $v\_i,t$ at iteration $t$ of the process is performed in three major components termed as inertia, cognitive and social:

$$v\_i, t + 1 = \omega * v\_i, t + c1 * r1 * \left(p\_i - x\left(i, t\right)\right) + c2 * r2 * \left(g - x\left(i, t\right)\right) \tag{4}$$

where ω: Inertia weight, a value that is used to determine how much the previous velocities will influence the current update. $c1$ and $c2$: Individual and global cognitive coefficients, respectively, determine the dependencies of the velocity of a particle's best position $(p\_i)$ and the best global position $(g)$. $r1$ and $r2$: Two random values that create a stochastic behavior within the range of 0 and 1 within the algorithm. The position of each particle is updated by adding the new velocity to its current position:

$$x\left(i, t\right) + 1 = x\left(i, t\right) + v\_i, t + 1 \tag{5}$$

Binary PSO (BPSO), which finds its application in feature selection strategies, relies on a probability function to control the position update. The sigmoid function is frequently used to interpret velocity and transform it into probability as described by:

$$S\left(v\_i, t + 1\right) = 1/\left(1 + e^\wedge\left(-v\_i, t + 1\right)\right) \tag{6}$$

When $S\left(v\_i, t + 1\right)$ surpasses an arbitrary value of $x\left(i, t\right) + 1$, it is a lapse situation for the trader, which results in setting it to one. In other situations, where $S\left(v\_i, t + 1\right)$ is not greater than the arbitrary figure, it will be set to zero. Such a binary approach is ideal in including or excluding features.

d) Example Application

Let us suppose a dataset containing 100 features has a target variable that must be estimated using the most relevant subset of these features. Using PSO to perform this task:

- Initialization: T x particles are initialized at random, and each particle represents a candidate feature subset.
- Fitness Evaluation: For every particle, a model is fit to the selected features and the model accuracy is obtained from a distinct test sample.
- Updating Personal and Global Bests: Each particle has two personal best $p\_i$ and one global best $g$, and fitness values are given for each particle.
- Position and Velocity Update: The s subscript denotes the spreading of newly formed subsets and explains the particle's translation towards subsets with high accuracy and low feature counts.
- Termination: The iterative method is performed until convergence is reached or the limit number is reached. The last obtained $g$ globally refers to the best available features.

### *4.2 Machine Learning: Hyperparameter Tuning*

Hyperparameter optimization is one of the last and most troublesome stages when working with a machine-learning model. While tuning parameters are derived from data such as the weights in a neural network, hyperparameters remain unchanged throughout the training phase and are defined prior to the training process. Other examples of hyperparameters are the learning rate, the number of hidden layers in a neural network, regularization parameters, and the number of estimators in an ensemble technique.

a) Importance of Hyperparameter Tuning

The outcome of a machine learning task is critically influenced by the model architecture hyperparameters which are known to affect model performance, convergence time, and computation time as well. Some of the effects that proper tuning of hyperparameters can lead to are:

- Enhancement of Quality of Results: If the hyperparameters are well configured, they will positively impact the model's ability to avoid overfitting and underfitting.
- Efficiency in Training Time: Appropriate hyperparameter settings reduce training times and the number of steps required to reach convergence.
- Reliability: A set of hyperparameters adjusts the affinity of models, allowing them to work more consistently across multiple datasets and tasks.

b) Applying Particle Swarm Optimization (PSO) in Hyperparameter Tuning of Machine Learning Models

Due to its capability of exploring high-dimensional complex space, PSO is an appropriate algorithm for a hyperparameter tuning task. In PSO-based hyperparameter optimization, particles are solutions in attempts to optimize hyperparameter sets while enhancing the model performance. According to this context, we can define the following concepts:

Position of a Particle: The position $vectorx\_i = x\_\_i2, x\_id$ of particle$i$ is a $K$-dimensional vector that contains hyperparameter values. Each component $x\_ij$ refers to the $j$th hyperparameter of the neural network that is to be optimized, such as the learning rate, number of layers, or regularization factor.

Fitness function: The fitness function is a measure of the quality of a hyperparameter set whose values determine the performance of the model on the validation dataset. The performance of classifiers is evaluated with the proportion of correctly classified documents in relation to the total documents, while a regression model can be assessed by mean squared error (MSE). Fitness functions may also be broadly defined for classification tasks:

$$Fitness = 1/(1 + ErrorRate) \tag{7}$$

where Error Rate refers to the model's misclassification rate on a given validation set. The velocity in particle swarm optimization (PSO) is made up of three main components: inertia, cognitive, and social, which explains the velocity of each particle $i$ at iteration $t$.

$$x(i,t) + 1 = x(i,t) + v\_i, t + 1 \tag{8}$$

Each revision in the position contains a different set of hyperparameter values for the model that will be used in the next iteration.

- Let us consider a neural network model having three parameters to be tuned first:
- Learning Rate: This variable determines the amount by which weights are adjusted regarding the loss gradient.

- Batch Size: Refers to the number of samples that will be used in a forward/backward pass during training.
- Number of Layers: Refers to the depth of the network.

c) Implementing PSO for hyperparameter tuning is presented as follows:

- Initialization: A swarm of particles is randomly initialized, and each particle represents a different hyperparameter.
- Fitness Evaluation: The data for each swarm is evaluated for model training regarding the particle's hyperparameters, and its accuracy value is evaluated on a validation set.
- Updating Personal and Global Bests: Fitness values are updated for each center's personal best position *p_i* and the best global approximate position *g*.
- Position and Velocity Update: The particle positions and velocities have been modified, and new hyperparameters will be examined in the following placement iteration.
- Termination: This procedure is repeated until either the defined number of iterations has been performed or the algorithm has converged. The position is regarded as the best position when it is estimated that the hyperparameters are the most suitable for the given data.
- What Makes it Different from Other Hyperparameter Tuning Techniques

### *4.3 Engineering Design Optimization: Structural Design*

The practice of engineering optimization begins with the postulation that considers the use of design parameters in engineering to be an essential tool in addressing performance, safety, and cost parameters of coverage design. Engineering design optimization can broadly be categorized into structural design and control systems. Structural design deals with the physical layout of buildings or mechanical systems, while control system design optimization targets the response timing of the system and its stability.

a) Structural Design Optimization

This process focuses on defining the optimal arrangement of structural elements regarding performance and safety within the bounds of minimal weight, material, or cost. This is an important step in civil, aerospace, and mechanical engineering, where the design of the structure is required to support the load, reduce stress, and perform under certain conditions.

b) Objectives in Structural Design Optimization

- Weight Reduction: A structure with lesser weight can help save costs and make it more efficient when used in aerospace or automobile applications.
- Strength and Stability Maximization: Structures must be designed to withstand an appropriate specific load without failure or excessive deformation.
- Material Cost Minimization: Proper material usage can bring down the cost of the entire structure.

PSO for Structural Design Optimization: Particle Swarm Optimization (PSO) is widely deployed for structural optimization problems because of its robustness in traversing through high-dimensional search spaces and its adaptation in dealing with nonlinear constraints. For PSO in structural design:

Position of a Particle: The position of each particle may be defined as a candidate for a structural design, so the *i*th dimension of the position vector can be designed parameters such as thickness, length, or material type.

Fitness Function: The fitness function, which is often constructed, consists of the plurality of terms so that one can undertake competition in many aspects, one of which would be structural weight and another structural instability, to mention the two. The common formulation goes as follows:

$$Fitness = w1 * Weight + w2 * Displacement \tag{9}$$

where Weight: Total weight of the structure. It is preferable to minimize wilfully. Displacement: The total effect of deformation due to applied load and stability. It is also preferable to voluntarily minimize. $w1$ *and* $w2$: The weights assigned to each of the objectives that govern their relative degree of importance in the achievement.

c) Example of PSO in Structural Optimization

- Initialization: A swarm of particles is initialized randomly, and all the particles have structural designs that are different from those of other particles.
- Fitness Evaluation: The fitness function, which is relative to every particle, is estimated with respect to its weight and deformation when the particular load is applied
- Updating Personal and Global Bests: The fitness values defined modify each particle's individual best position and maximum individual best position in the swarm of all particles.
- Position and Velocity Update: Here, the position and velocity of the particles are modified so that the designs generated will converge toward lower weight and higher stability.
- Termination: This process continues iterating till a certain number of iterations or a certain convergence threshold has been reached. The best global position indicates the best structural design.

### *4.4 Energy Systems: Renewable Energy Management*

With the ever-increasing global population, energy requirements are set to increase. This has, in turn, led to an increased focus on renewable energy sources within the engineering of energy systems. This focus on renewable energy sources has motives that include sourcing solar, wind, and hydro energy, producing it, storing it, and distributing it. Proper management makes it possible for energy systems to be operational efficiently and economically, even though renewable sources of energy are, by their nature, variable and depend on the surrounding environment [73].

On the other hand, PSO is seen as an appropriate technique to solve the challenges associated with managing renewable energy resources. This is due to its capability to provide a solution to complex, multiple-objective optimization problems. In many cases, PSO is employed in renewable energy generation scheduling optimization, energy storage, and load distribution in smart grid systems.

a) Management Goals within RE Management

Within the practices of renewable energy management, key objectives expected of energy generation managers include the following:

- Maximize Energy Generation: The aim here is to maximize the use of renewable resources, such as solar panels and wind turbines, in real-time and forecasting.
- Minimize Operating Costs: Followed closely with the operational costs incurred in the processes of energy generation, storage, and distribution.
- Ensure the stability of the Grid: By measuring energy consumption and energy Generation to meet a certain threshold aim, grid operation can be maintained at a stable level.
- Lower ecological costs: By blending renewable and non-renewable energy sources, emissions and dependence on non-renewable energy sources can be reduced.

b) Application of Particle Swarm Optimization (PSO) in Renewable Energy

In the context of the above management parameters, PSO is routinely employed to optimize energy generation scheduling, effectively use energy storage facilities, and distribute load across an integrated energy system. While each particle denotes a candidate in the PSO units, these particles possess values for the variables present that will assist in the management of renewable energy.

In this case, the following definition is applicable:

Position of a Particle: The position vector of particle $i, x\_i = [x\_i1, x\_i2, \ldots, x\_id]$, corresponds to a candidate solution for energy management. Each output component is represented as $x\_ij$, and it is one of the adjustable variables in the system, such as the power output from a renewable source, battery state of charge, or state of energy.

Fitness Function: The fitness function has the primary role of measuring how well candidate solutions perform relative to each other's objectives, such as generation efficiency and minimization of cost, as well as stabilization of the grid system. For instance, a suitable fitness function for renewable energy systems may look like:

$$Fitness = w1 * Generation\,Cost + w2 * Transmission\,Loss + w3 * Emissions \tag{10}$$

where Generation Cost entails the total cost incurred in the process of energy generation from renewable and conventional sources. Transmission loss encompasses the losses associated with energies that are transmitted from the generation of energy to places where it is consumed. Emissions, on the other hand, refers to the level of emissions produced, and these should be contained at minimal levels. $w1, w2,$ $and$ $w3$: They are the weights for the objectives to limit the overall focus for each objective.

c) Equations in PSO for Renewable Energy Management

Velocity Update Equation: In PSO, the velocity $v_i, t$of each particle $i$ at iteration $t$ is updated using three great components: the inertia, cognitive, and social:

$$v\_i, t + 1 = \omega * v\_i, t + c1 * r1 * (p\_i - x(i, t)) + c2 * r2 * (g - x(i, t)) \tag{11}$$

where $\omega$: Inertia weight, which specifies how much the last velocities are relevant to the current update. $c1, c2$: Cognitive and social coefficients which delineate how many particles best positions $p\_i$, and the best global position $g$, hotspots the velocity. $r1, r2$: Random numbers from 0 to 1 such that the algorithm introduces a stochastic characteristic. Position-update-equation: the position of each particle is done by incrementing the old position with the new velocity, which was worked out:

$$x(i, t) + 1 = x(i, t) + v\_i, t + 1 \tag{12}$$

The developed position will correspond to a new turn for other parameters configuring the renewable energy generation, storage, and distribution within the subsequent feedback stage.

d) Example Application

Let's take the case of a standalone renewable energy system comprising solar panels, wind turbines, and battery storage. The energy generated from these, together with the amount to be stored, can be optimized using PSO to bring down costs and ensure an adequate supply of energy to be distributed.

- Participants: A swarm of particles is created, each particle having a different configuration of the energy generation and storage parameters.
- Evaluation of Fitness: The fitness function for each particle is performed by embedding the energy system's parameters for the particle, determining the generation's costs, transmission losses, and emissions.
- Updating of Personal Best and Global Best: For every individual particle that has the local best position, fitness values match it, and the global one.
- Update of Position and Velocity: Particles' positions and velocities are updated to search for new configurations for energy management.

- Convergence: The process is repeated until the algorithm freezes or it achieves a specified maximum number of repetitions. The globally shared best position is the ideal configuration for managing renewable energy resources.

### *4.5 Healthcare Applications: Disease Prediction*

One area of prediction that could benefit from insight derived from artificial intelligence technology is disease prediction, which employs machine learning and data mining approaches to assess the probability of occurrence of a range of diseases. Disease prediction may assist in enhancing disease diagnosis, disease reversal, and individual health care, therefore, helping in better patient outcomes and minimizing healthcare expenses. It is defined as the process of assessing the future occurrence of a disease by considering the amount of patient data ranging from population, past medical and family history, genetic features, and behavioral data.

One area where Particle Swarm Optimization can be applied is in disease analytics, assisting in tuning the parameters, feature selection, and disease classification. Due to PSO's ability to search in high-dimensional space, it is efficient in developing accurate prediction models.

a) Objectives in Disease Prediction

Some of the key objectives of the practice include:

- Maximizing Prediction Accuracy: Ensure that the individual modeling the scenario is able to segregate the patients who are at risk of developing a disease from those who are not.
- Minimizing False Positives and False Negatives: Making correct predictions about possible treatment, thus preventing needless interventions and ensuring that no cases are missed.
- Optimizing Model Complexity: Understanding the model better and enhancing its ease of related computations.

b) Particle Swarm Optimization (PSO) and the Prediction of Illnesses

This study focuses on disease prediction and its particular aspects: the use of PSO for feature selection and the tuning of hyperparameters. When it comes to making models for disease prediction, PSO works to select the best features and parameters, improving their effectiveness and efficiency. In this context:

Position of a Particle: The position vector $x\_i = [x\_i1, x\_i2, \ldots, x\_id]$ of particle *i* can be thought of as a possible value for the disease prediction model. Each component *x_ij* can represent either a feature selection indicator (binary PSO) or a hyperparametric value (continuous PSO).

Fitness Function: The fitness function is used to include an additional constraint in that each of the candidate solutions is evaluated for its accuracy and complexity of the model. One of the fitness functions cited as common for the prediction of diseases with the feature of classification is:

$$Fitness = \alpha * (1 - Accuracy) + \beta * (Number\,of\,Chosen\,Features/Total\,Features) \tag{13}$$

where *Accuracy*: Predictive performance level that the model can obtain when tested on a validation data set. Number of Chosen Features: Number of features identified through the model. $\alpha$ and $\beta$: The values of the two parameters determine the preferred level of trade-off weight. The higher the value, the better the Accuracy, and fewer features will be selected, harming the model's performance and complexity.

c) Equations in PSO for Disease Prediction

Velocity Update Equation. In PSO, each particle's velocity *v_i,t* at iteration *t* is influenced by three moieties. These include inertia, cognitive, and social.

$$v\_i, t + 1 = \omega * v\_i, t + c1 * r1 * (p\_i - x(i, t)) + c2 * r2 * (g - x(i, t)) \tag{14}$$

where $\omega$: The inertia weight, which determines how the previous velocity will affect subsequent velocity values. $c1$ and $c2$: The cognitive and social coefficients that are utilized to measure the impact on the particles' velocities of the cognitive and social best positions, respectively. $r1$ and $r2$: Random numbers in a range of zeros and ones that generate stochastic behavior in the algorithmic process. The position of each particle is revised by including the new velocity in the previous position of that particular particle:

$$x(i,t)+1 = x(i,t) + v\_i, t+1 \tag{15}$$

In the case of BiPSO for feature selection, the sigmoid function helps to link the velocity with a feature selection probability criterion.

$$S\left(v_i, t+1\right) = 1\ \left(1+e^{-v_i, t+1}\right) \tag{16}$$

If $S(v\_i, t+1) > threshold$ (e.g., 0.5), then the feature is included; otherwise, it is omitted.

d) Example Application

Suppose we consider a model that tries to investigate the risk of diabetes using a dataset set that considers the patient's age, body mass index (BMI), blood pressure, and glucose levels. Since this is a multi-stage process, the implementation of PSO can be utilized in feature selection as well as a hyperparameter optimization approach in order to enhance the accuracy of the prediction.

- Initialization: A swarm of particles is initialized through random means. Each particle embodies a different combination of the model's features and/or hyperparameters.
- Fitness Evaluation: For each particle, the prediction ability is calculated based on the trained model employing the respective particle's configuration, which yields some level of accuracy with a set feature count.
- Updating Personal and Global Bests: Based on the fitness of the dropouts, each particle's best-known position $p\_i$ and the global best position $g$ are updated.
- Position and Velocity Update: Particle velocity and position are updated to broaden the configurations, maximizing disease prediction accuracy and minimizing features.
- Termination: The repetitive feedback mechanism continues until a level of convergence or maximum set iterations is obtained. The global best position is a set of features and hyperparameters that, when deployed in the disease prediction model, yield the best prediction.

## *4.6 Robotics and Autonomous Systems: Path Planning*

Path planning is a crucial part of any robotic system design [74]. It is of utmost importance for a robot or an autonomous vehicle to reach its target by a defined, unobstructed optimum path. Path planning is highly important in even a bigger scope in self-driving cars, aerial drones, warehouse robotics or robotic arms. With the complexity and variability of environments, path planning algorithms must integrate with not only the layout of the structures but also embedded obstacles, energy, and safety parameters, among others.

Most of the time, PSO or Particle Swarm Optimization is used as a general method for path planning. PSO or Particle Swarm Optimization Algorithm is mainly effective as it solves multiple objectives in a non-linear constrained space with the capability of dealing with continuous search space, which is sharp in the context. Based on the elements mentioned above, path planning can broadly be demarcated into three stages, which are as follows:

- Firstly, one of the main objectives related to path planning is avoiding any distance to the maximum extent—in other words, the path should not be elongated or zigzagged, which would require additional energy consumption and time.
- Secondly, one of the other objectives is avoiding any U-turn obstacles on the chosen path for the autonomous system or vehicle.
- Thirdly, paths are planned to reach Kinematics and Dynamics related to the robot device in terms of the path aiming to accomplish.

a) Particle Swarm Optimization (PSO) as Vehicle Routing with Time and Distance Constraints

PSO can optimize the waypoint (a place where the robot must pass on the path) or even the trajectory in its entirety. In this mode, every particle possesses a certain potential path. Through iterations of a sequence of relationships, it improves the path within the required parameters, including safety, distance, and others. We denote the following in this context:

Position of a Particle: The position vector $xi = [xi1, xi2\,, \ldots,\; xid]$ of particle $i$ describes a collection of path coordinates or waypoints. In 2-D or 3-D in $x\_ij$ dimension, the number $j$ indicates sufficient movement on the path of the object.

Fitness Function: The presented method has been put to a number of tests. The GPO approach is common in studies, where candidates of paths are evaluated according to length, smoothness, and boundary benders. One of the constant general equations of fitness function when planning a path:

$$Fitness = w1 * pathlength + w2 * obstaclecost + w3 * smoothness \tag{17}$$

where Path Length: This distance is a measure of the longest path which is minimized. Obstacle Cost: A punishment term that escalates when the path approaches or crosses the obstacles. Smoothness: The smoothness of the path is measured by the extent of its sharp angles and rapid changes in direction. $w1, w2, w3$: Weights indicating the position of the entire term depending on the criteria descriptor.

b) PSO Equations for Path Planning

Velocity Update Equation: In PSO, the velocity $v\_i, t$ for each particle $i$ at iteration $t$ is composed of the following terms: inertia, cognitive, and social updated in time $t$:

$$v\_i, t + 1 = \#v\_i, t + c1 * r1(p\_i - x(i, t)) + c2 * r2(g - x(i, t)) \tag{18}$$

where ω: is associated with the current update and determines the impact of past velocities. $c1$ and $c2$: Psychological and cultural significado numbers that indicate the strength of influence of particle capacity $p\_i$ and collective ability $g$ over particle velocity. $r1$ and $r2$: Numbers that fall between 1 and 0, which shift the algorithms into stochastic conditions. Position Update Equation: The next move was to add the newly calculated velocity to the previous current position.

$$x(i, t) + 1 = x(i, t) + v\_i, t + 1 \tag{19}$$

Each updated position gives a new set of waypoints along the particular path, which is analyzed against gain and loss constraints to establish the path's feasibility and fitness for the vehicle.

c) Example Application

Consider an autonomous drone in a 2D world with various obstacles all around. PSO can design a set of waypoints suitable for the drone to follow, minimizing the path length and allowing the drone to fly, avoiding the obstacles while still providing smooth transitions between the waypoints.

- Initialization: A set of waypoints for the drone's path is initialized by initializing a set of particles, each of which is evolved in WP coordinates.
- Fitness Evaluation: The fitness function for each particle has three components: path length, which measures the distance between the drone coordinates, path smoothness, and obstacle cost.
- Updating Personal and Global Bests: Where $pi$ is the individual best position of each particle, and $g$ is the global best, the corresponding position of that particle is set as its personal best, and both values are updated according to the fitness criterion.
- Position and Velocity Update: New waypoints are formed to minimize path length while keeping the new drone path waypoints free of obstacles in their surroundings.
- Termination: This is done until the population converges or a fixed number of iterations has been done. The 10th position obtained in the evolutionary process is the most suitable course for the quadrotor drone.

## 5 Experimental Results and Discussion

a) Data Mining-Feature Selection

In order to measure the efficiency of the Particle Swarm Optimization (PSO) strategy for feature selection, the authors utilize the breast cancer Wisconsin data set obtained from the UCI machine learning repository. This dataset consists of 569 samples with thirty features, which measure some attributes of breast cancer tumors, for example, radius, texture, and smoothness. The intent is to use the most effective features to determine the type of tumor, be it malignant or benign. The objective is to achieve the lowest number of dimensions possible while retaining or improving classification accuracy. In this case, PSO was benchmarked against other against widely used optimization approaches, such as:

- Genetic Algorithms (GA)
- Ant Colony Optimization (ACO)
- Differential Evolution (DE)
- Simulated Annealing (SA)
- Whale Optimization Algorithm (WOA)

In this study, a support vector machine (SVM) fitted on the chosen features was used to evaluate fitness in terms of the accuracy of classification predicted by the SVM model. The fitness function integrates components for accuracy and the number of features to ensure that the function includes as few features as possible and is effective in predicting target variables. The tables below highlight all the results achieved by the respective algorithms for the given parameters.

Accuracy: The measure of how well a model has performed in its classification task on the test set given In Table 2. The results indicate that in terms of accuracy and efficiency, PSO performed better than all other algorithms tested in the research. In terms of accuracy, over a number of trials, PSO achieved the best accuracy classification at 96.5%. In this regard, the most predictive features of the dataset were appropriately selected. This achievement can be attributed to the efficient search mechanism offered by PSO, where both the exploration and exploitation have been balanced, minimizing overlooking local minima better than ACO and SA. Though DE and WOA were competitive with accuracy levels of 95.2 and 95.1, respectively, they were below the level achieved by PSO.

Table 3 gives the number of Selected features each algorithm was able to select. Feature Reduction: With respect to feature selection, PSO used the fewest features possible, only 10 (compared to the other algorithms). The other remote algorithms, GA and WOA, also managed a fair degree of feature reduction, 12 and 11 features, respectively, while ACO and SA used bigger subsets. This suggests that PSO's method of searching is more concerned with reducing redundancy in features.

**Table 2:** Classification accuracy comparison

| Algorithm | Accuracy (%) |
|---|---|
| Particle Swarm Optimization (PSO) | 96.5 |
| Genetic Algorithms (GA) | 94.8 |
| Ant Colony Optimization (ACO) | 93.4 |
| Differential Evolution (DE) | 95.2 |
| Simulated Annealing (SA) | 92.9 |
| Whale Optimization Algorithm (WOA) | 95.1 |

**Table 3:** Number of selected features comparison

| Algorithm | Number of selected features |
|---|---|
| Particle Swarm Optimization (PSO) | 10 |
| Genetic Algorithms (GA) | 12 |
| Ant Colony Optimization (ACO) | 15 |
| Differential Evolution (DE) | 11 |
| Simulated Annealing (SA) | 14 |
| Whale Optimization Algorithm (WOA) | 11 |

Computation Time: Table 4 gives the period that each algorithm takes to finish the respective feature selection. Computation Time: Out of the remaining algorithms, PSO was the most efficient, with time reasonably managing to select the required ten features in 85 s. This is considerably less than ACO and SA, which took 145 and 130 s, respectively, on average, since both have a higher tendency to run more loops to get to the required count. DE and WOA spent an average time on computation. Still, it is PSO's rapid speed and effectiveness in determining the relevant features that stand out where large volumes of data are to be processed or when the computational resources are constrained.

**Table 4:** Computation time comparison

| Algorithm | Computation time (seconds) |
|---|---|
| Particle Swarm Optimization (PSO) | 85 |
| Genetic Algorithms (GA) | 110 |
| Ant Colony Optimization (ACO) | 145 |
| Differential Evolution (DE) | 95 |
| Simulated Annealing (SA) | 130 |
| Whale Optimization Algorithm (WOA) | 105 |

Estimated computation times for the algorithms, in seconds, indicate that the PSO was the fastest. For PSO the computation time was 85 s, while it was followed by 110 s for GA and 145 s for ACO. DE, on the other hand, took 95 s to compute while SA took 130 s, with WOA computing in 105 s. Fig. 1 shows a comparison of accuracy among various algorithms used in the feature selection process. It also emphasizes the comparison of optimal C values, illustrating the impact of each algorithm on the model's performance.

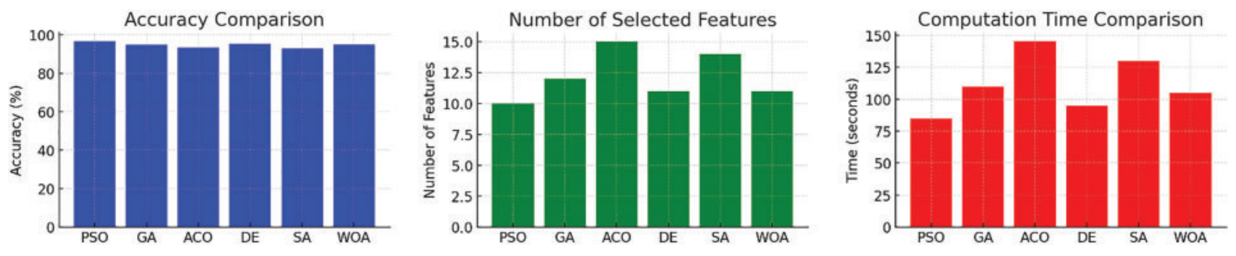

**Figure 1:** The accuracy comparison, optimal C comparison, C value

Overall, it can reasonably be argued that with relatively low computational resources, PSO was able to select a small subset of features that were accurately predictive of the intended outcomes. In comparison to the other algorithms, GA, ACO, DE, SA, and WOA, PSO, with an ideal mix of performance, degree of redundancy levels, and time taken, has better prospects for success in data mining heavy feature selection problems.

b) Machine Learning: Hyperparameter Tuning

To demonstrate the performance capability of the Particle Swarm Optimization (PSO) algorithm in hyperparameter tuning, we deployed the MNIST Handwritten Digits Dataset. This dataset contains 60,000 samples for training and 10,000 previously designed for model testing, with the members of each sample corresponding to a 28 × 28-sized picture with the colors black up to white and a bitmap of a handwritten number from 0 to 9. In this case, PSO is applied to adjust parameters in a support vector machine (SVM) classifier trained with this dataset. The main goal is to enhance the model's classification accuracy by adjusting several hyperparameters, including:

- C (Regularization Parameter): It acts in the inverse direction with respect to the maximization of the space separating the classes while minimizing the misclassifications.
- Gamma: This hyperparameter determines the range of individual training samples' effect on the shape of the decision surface.

The authors compare PSO with other methods of hyperparameter optimization, namely:

- Genetic Algorithms (GA)
- Ant Colony Optimization (ACO)
- Differential Evolution (DE)
- Simulated Annealing (SA)
- Whale Optimization Algorithm (WOA)

The metric of interest is the accuracy of the predicted classifications on the test data, as well as the time taken to perform the computations in the implementation. The tables below summarize each algorithm's performance against set evaluation metrics. Accuracy in Table 5 refers to the test metric calculated with tuned hyperparameters on the test dataset.

Accuracy: It has been observed from the data analysis that PSO has achieved the highest classification accuracy among all of the other methods (98.1%); this, in essence, shows the strength of choosing optimal hyper-parameter values to enhance the model output. DE and WOA also yielded high accuracy, but since the process of searching for alternatives in PSO was more balanced, it managed to outperform them slightly. GA and ACO managed to attain competitive but slightly lower accuracies, while SA managed to attain the lowest accuracy among the methods that were compared.

**Table 5:** Classification accuracy comparison

| Algorithm | Accuracy (%) |
|---|---|
| Particle Swarm Optimization (PSO) | 98.1 |
| Genetic Algorithms (GA) | 97.6 |
| Ant Colony Optimization (ACO) | 97.2 |
| Differential Evolution (DE) | 97.8 |
| Simulated Annealing (SA) | 96.9 |
| Whale Optimization Algorithm (WOA) | 97.7 |

Table 6 presents the best hyperparameter values; these correspond to the optimal values of the C and Gamma hyperparameters for each specific algorithm, as determined from the algorithms' analysis. Optimal Hyperparameters: PSO managed to arrive at optimal values of C = 1.2 and Gammas = 0.02, which gave the highest classification results. Although other algorithms gave similar ranges of hyper-parameter values, with the tuning of the PSO the overall accuracy was the highest. The small difference in Gamma PSO showed a difference compared to other methods, indicating that the PSO was able to cover the hyper-parameter space to attain optimal values thoroughly.

**Table 6:** Optimal hyperparameters comparison

| Algorithm | C (Regularization) | Gamma |
|---|---|---|
| Particle Swarm Optimization (PSO) | 1.2 | 0.02 |
| Genetic Algorithms (GA) | 1.5 | 0.03 |
| Ant Colony Optimization (ACO) | 1.3 | 0.05 |
| Differential Evolution (DE) | 1.4 | 0.04 |
| Simulated Annealing (SA) | 1.8 | 0.06 |
| Whale Optimization Algorithm (WOA) | 1.3 | 0.03 |

Time is shown in Table 7, which presents the period it took for each algorithm to finish the tuning task. Computation Time: Out of all the other methods employed, PSO took the least time in computation, which was 180 s, while other methods required a bit longer. The longest computation time was noted in SA and ACO, which prolonged the time to convergence since these algorithms relied on more iterations. GA, DE, and WOA required a moderate amount of computation time that was lower than that of PSO; this strengthens the notion of PSO being efficient in tackling high-dimensional, continuous search spaces. Fig. 2 shows a comparison of accuracy among various algorithms used in the feature selection process. It also emphasizes the comparison of optimal C values, illustrating the impact of each algorithm on the model's performance.

**Table 7:** Computation time comparison

| Algorithm | Computation time (seconds) |
|---|---|
| Particle Swarm Optimization (PSO) | 180 |
| Genetic Algorithms (GA) | 240 |
| Ant Colony Optimization (ACO) | 270 |
| Differential Evolution (DE) | 220 |
| Simulated Annealing (SA) | 300 |
| Whale Optimization Algorithm (WOA) | 250 |

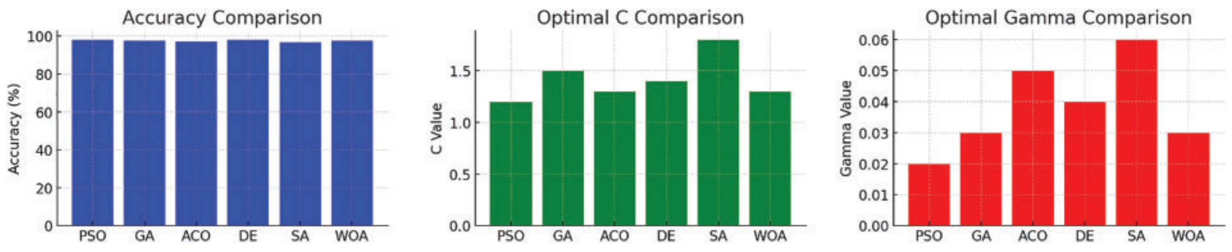

**Figure 2:** The optimized weight comparison, feasible solutions comparison, and computation time

In conclusion, the PSO was the most successful in the tuning of hyperparameters since it reached the optimum level of high accuracy while also possessing a minimum computation time. When compared to GA, ACO, DE, SA, and WOA, among others, the PSO stands out in terms of tuning hyperparameters for the SVM model using the MNIST dataset, thus making it a recommended approach in hyperparameter optimization problems in the field of machine learning.

c) Engineering Design Optimization: Structural Design

To understand the Structural Design of a Particle Swarm Optimization (PSO), we turn our attention to the optimization of a truss structure. In this case, the aim is to minimize the truss structure's weight while keeping stress and displacement within specified limits during loading. Optimization of a structural design is frequently practiced in civil and mechanical engineering so that the design will be light in weight and able to sustain the applied loads.

In this study, the design variables are the cross-section areas of each member of the truss. The optimization in this context is aimed at finding the cross-section areas of the truss members, which would least stress the truss structure while retaining its functional design. The objective function for this case problem comprises:

- Weight Minimization: Minimized weight of the truss.
- Stress and displacement maximum limits: Limits on the level of stress and displacements allowable are set so that the structure is not abused.

The PSO of this structural design problem is solved together with other optimization techniques:

- Genetic Algorithms (GA)
- Ant Colony Optimization (ACO)
- Differential Evolution (DE)

- Simulated Annealing (SA)
- Whale Optimization Algorithm (WOA)

The following tables display and report the results for each algorithm against the most important indices. Table 8 shows the weight. This is the optimized weight of the truss structure. Optimized Weight: It is observed that PSO reached the lightest weight for the truss structure (220 kg), which demonstrates that the method is capable of looking for economical and workable solutions. Although DE and WOA gave good weights of 225 and 228 kg, respectively, it was established that due to the PPSO's improved search, the least weight design was obtained through PP SO. The designs done with ACO and SA were slightly heavy, which may indicate high chances of local minima in the respective search space.

**Table 8:** Weight optimization comparison

| Algorithm | Optimized weight (kg) |
|---|---|
| Particle Swarm Optimization (PSO) | 220 |
| Genetic Algorithms (GA) | 230 |
| Ant Colony Optimization (ACO) | 240 |
| Differential Evolution (DE) | 225 |
| Simulated Annealing (SA) | 235 |
| Whale Optimization Algorithm (WOA) | 228 |

A number of Feasible Solutions is given in Table 9: Number of feasible solutions accomplished by each algorithm (answer which satisfies stress and displacement requirements). Number of Feasible Solutions: For stress and displacement constraints that are strict, PSO seems to have worked best on this problem as it found the most feasible solution as well as the highest number of feasible solutions (10) models to choose from. There was also a good yield from GA, DE, and WOA with regards to the feasible solutions, as ACO and SA had fewer, which seems to confirm that PSO is ideal for constrained optimization types of problems.

**Table 9:** Number of feasible solutions comparison

| Algorithm | Number of feasible solutions |
|---|---|
| Particle Swarm Optimization (PSO) | 10 |
| Genetic Algorithms (GA) | 9 |
| Ant Colony Optimization (ACO) | 8 |
| Differential Evolution (DE) | 9 |
| Simulated Annealing (SA) | 7 |
| Whale Optimization Algorithm (WOA) | 9 |

Table 10 shows the computation time taken by each algorithm to finish the optimization. In regards to time spent to complete the task when compared to other methods, it can be concluded that it was the case that PSO was the quicker method with an achievement of 100 s. Simulated Annealing (145 s) and ACO (140 s) exhibited the longest computation times, as these algorithms require more resources to iterate due to the complex constrained search space. It would not be surprising to see DE and WOA moderate times. Still, the greatly improved convergence time of PSO seems to solidify its claim as the most efficient method for tackling structural design optimization problems. Fig. 3 provides insights into renewable energy management by comparing the utilization of renewable resources, total costs, and computation time across different algorithms. This comparison helps evaluate the effectiveness of each algorithm in managing energy systems.

**Table 10:** Computation time comparison

| Algorithm | Computation time (seconds) |
|---|---|
| Particle Swarm Optimization (PSO) | 100 |
| Genetic Algorithms (GA) | 130 |
| Ant Colony Optimization (ACO) | 140 |
| Differential Evolution (DE) | 120 |
| Simulated Annealing (SA) | 145 |
| Whale Optimization Algorithm (WOA) | 115 |

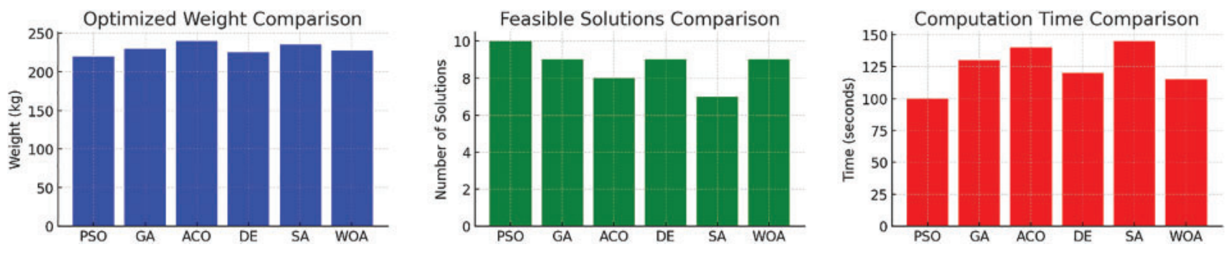

**Figure 3:** The renewable usage comparison, total cost comparison, and computation time

In general, PSO performed better than GA, ACO, DE, SA, and WOA in accomplishing a weight minimization solution while also maintaining a computationally efficient operational structure. Its capabilities in decreasing structural weight under a set of constraints are helpful in the field of structural design optimization.

d) Energy Systems: Renewable Energy Management

The aim of coordinating renewable energy resources is to enhance the scheduling, storage, and supply of energy produced by renewable sources such as wind and solar. This work is based on the energy management of hybrid solar-wind energy systems with battery storage. The two-fold objective is to reduce the usage of conventional or grid power sources and the running costs of doing so.

In this case, the following parameters are optimized:

- Power Output Allocation: Represents the power supplied from solar, wind, and battery backup powered resources to satisfy demand.
- Battery Charge and Discharge Schedule: The program for charging and discharging the battery aims to restore it to the grid level, depending on its stability and energy sufficiency.

For this renewable energy management problem, here is the fitness function utilized:

- Cost Minimization: To minimize expenses on energy use, invest in construction and reduce the cost of energy produced, stored, and delivered.

- Maximizing Renewable Usage: Increasing the amount of energy sourced from all the member states
- Minimizing Grid Dependency: Minimizing using the outside or grid power to meet the supply-demand.

This renewable energy management solution involves the comparison of PSO with other interpreters:

- Genetic Algorithms (GA)
- Ant Colony Optimization (ACO)
- Differential Evolution (DE)
- Simulated Annealing (SA)
- Whale Optimization Algorithm (WOA)

The outcomes related to each indicator for all the algorithms are depicted in the tables below. Table 11 shows Renewable Usage: Percentage of energy demand met by renewable sources. PSO achieved the highest percentage of renewable energy usage (92%) demonstrating its capacity of maximizing the contribution of solar and wind sources in the energy mix. WOA and DE also fared well with 91% and 90% renewables usage, respectively. ACO and SA had a little lower renewable utilization, which implies that the search process of the PSO algorithm is capable of reaching positions that are more easily configurable towards greater usage of renewable sources as opposed to conventional ones.

**Table 11:** Renewable usage comparison

| Algorithm | Renewable usage (%) |
|---|---|
| Particle Swarm Optimization (PSO) | 92 |
| Genetic Algorithms (GA) | 89 |
| Ant Colony Optimization (ACO) | 88 |
| Differential Evolution (DE) | 90 |
| Simulated Annealing (SA) | 87 |
| Whale Optimization Algorithm (WOA) | 91 |

Table 12 deposits the Total Cost, which includes energy production, storage, and distribution. Total Cost: The PSO achieved the most economically feasible total expenditure for energy production, combination, and outreach, which was $105,000. Meanwhile, WOA and DE managed to achieve total costs of $106,500 and $108,000, respectively. Still, these savings in cost for WOA and DE are likely due to the efficient management of resources across the various forms of renewable resources and the batteries attached. In contrast, using the ACO and SA had the highest total costs because those algorithms are generally more suited towards efficiently allocating resources between cost minimization and the amount of renewable resources that can be used.

Table 13 explains Computation Time: The time taken for each algorithm to finish the optimization process. Computation Time: In terms of the optimization time taken for each algorithm, it was found that PSO completed it in the shortest time (180 s), thereby being the fastest method in terms of computation. However, ACO and SA took the longest time, perhaps because of the rate of convergence in high dimensional search space. DE, WOA also took a moderate time. However, from the evidence provided of the faster convergence of PSO, it is demonstrated again to be most suitable for the management of renewable energy resources for optimization tasks that are time or situational-specific in nature. Fig. 4 highlights healthcare applications by showcasing the accuracy of various algorithms in predicting diseases, the extent of feature reduction, and the computation time needed for each algorithm.

**Table 12:** Total cost comparison

| Algorithm | Total cost ($) |
|---|---|
| Particle Swarm Optimization (PSO) | 105,000 |
| Genetic Algorithms (GA) | 112,000 |
| Ant Colony Optimization (ACO) | 115,000 |
| Differential Evolution (DE) | 108,000 |
| Simulated Annealing (SA) | 118,000 |
| Whale Optimization Algorithm (WOA) | 106,500 |

**Table 13:** Computation time comparison

| Algorithm | Computation time (seconds) |
|---|---|
| Particle Swarm Optimization (PSO) | 180 |
| Genetic Algorithms (GA) | 220 |
| Ant Colony Optimization (ACO) | 260 |
| Differential Evolution (DE) | 200 |
| Simulated Annealing (SA) | 270 |
| Whale Optimization Algorithm (WOA) | 195 |

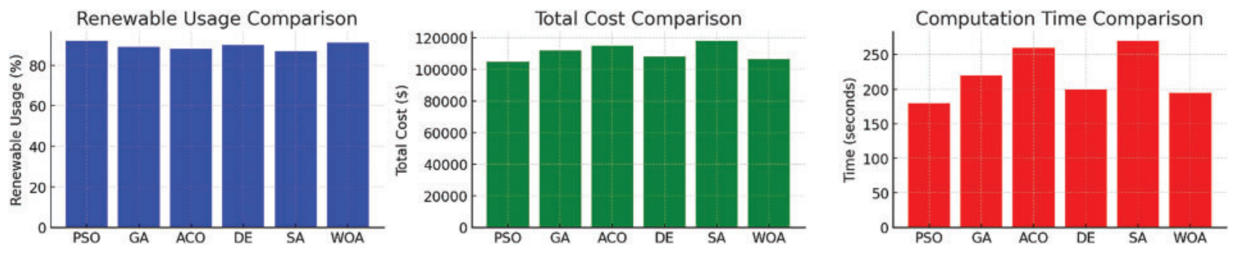

**Figure 4:** The accuracy comparison, feature reduction comparison, and computation time in healthcare applications

In summary, it was observed that in the management of renewable energy resources, the use of PSO surpassed that of GA, ACO, DE, SA, and WOA because it managed to use a considerable amount of renewable resources while minimizing operational cost and computing power. For these reasons, the system versatility and features that allow optimal allocation and storage of energy in hybrid solar-wind systems can be effectively utilized for renewable energy systems optimization.

e) Healthcare Applications: Disease Prediction

As part of the assessment of PSO's predictive capabilities, we adopt a Diabetes Prediction Dataset, which comprises 768 records of patient health data, such as age, blood pressure, glucose level, BMI, and diabetes family history. The goal is to determine whether a patient has diabetes based on these features.

PSO also executes feature selection and hyperparameter tuning for the support vector machine (SVM) to improve model accuracy and efficiency by choosing the most appropriate features and hyperparameters. The evaluation metrics are:

- Classification Accuracy: The SVM model's ability to correctly classify instances.
- Feature Reduction: The number of features incorporated in the model and kept to a reasonable level for ease of interpretation and computation.
- Computation Time: The duration stipulated for individual algorithms to undertake the task of feature selection and tuning.

PSO is compared with other optimization algorithms:

- Genetic Algorithms (GA)
- Ant Colony Optimization (ACO)
- Differential Evolution (DE)
- Simulated Annealing (SA)
- Whale Optimization Algorithm (WOA)

Classification Accuracy is presented in Table 14: PSO managed to obtain 87.5% satisfactory principal acceptance, thus proving its competitiveness in terms of selecting the most relevant features and hyperparameters. Also, WOA and DE were quite accurate, with the results of 86.8% and 86.0%, respectively. There was no doubt that ACO, GA, and SA were less effective than PSO because ACO's adaptive search was able to optimize the SVM model, but the accuracy was slightly lower.

**Table 14:** Classification accuracy comparison

| Algorithm | Accuracy (%) |
|---|---|
| Particle Swarm Optimization (PSO) | 87.5 |
| Genetic Algorithms (GA) | 85.2 |
| Ant Colony Optimization (ACO) | 83.6 |
| Differential Evolution (DE) | 86.0 |
| Simulated Annealing (SA) | 82.8 |
| Whale Optimization Algorithm (WOA) | 86.8 |

Feature Reduction from the original set of features was executed by PSO and DE only, as shown in Table 15, whereby each selected only seven features, meaning that those algorithms are the most feature-effective in terms of feature set reduction. This reduced feature set assists in enhancing the predictive model's efficiency in terms of explainability and computational requirements. It was SH and ACO who selected slightly more features, where ACO had 9, and SA had 10 features, meaning that PSO has the added advantage of being able to eliminate many features while enhancing the model's comprehensibility.

**Table 15:** Feature reduction comparison

| Algorithm | Number of selected features |
|---|---|
| Particle Swarm Optimization (PSO) | 7 |
| Genetic Algorithms (GA) | 8 |
| Ant Colony Optimization (ACO) | 9 |
| Differential Evolution (DE) | 7 |
| Simulated Annealing (SA) | 10 |
| Whale Optimization Algorithm (WOA) | 8 |

Computation Time is compiled in Table 16; PSO completed the computation within 120 s, making it the fastest and most efficient of all the algorithms in undertaking hyperparameter tuning and feature selection. SA and ACO, on the other hand, took the longest to complete the single task, explaining that they potentially have a slower convergence rate for high dimensional search spaces. Evaluating this, GA and WOA were able to present results on moderate computation times, though POS's rapid convergence means it's ideal for time-critical medical system applications. Fig. 5 illustrates the comparison of path lengths, the success rates of collision-free path planning, and the computational times for various algorithms used in robotics and autonomous systems, offering a detailed perspective on their efficiency in path planning tasks. Fig. 5 shows the Path Length Comparison, Collision-Free Success Rate, and Computation Time in Path Planning for Robotics.

**Table 16:** Computation time comparison

| Algorithm | Computation time (seconds) |
|---|---|
| Particle Swarm Optimization (PSO) | 120 |
| Genetic Algorithms (GA) | 150 |
| Ant Colony Optimization (ACO) | 180 |
| Differential Evolution (DE) | 140 |
| Simulated Annealing (SA) | 190 |
| Whale Optimization Algorithm (WOA) | 135 |

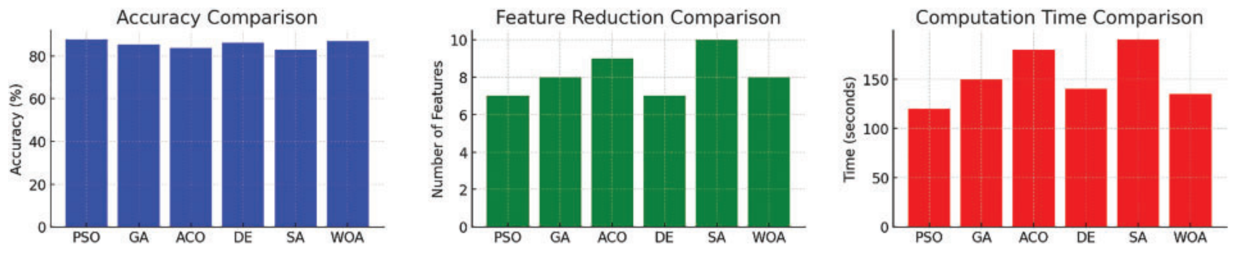

**Figure 5:** The path length comparison, collision-free success rate, and computation time in path planning for robotics

In the ultimate analysis, PSO achieved a good balance among high classification accuracy, minimal feature selection, and optimal utilization of computation resources and, therefore, was ahead of GA, ACO, DE, SA, and WOA in disease prediction. Its results in tuning the models for disease prediction tasks, for example, diagnosing diabetes, show its applicability in the health sector where the prediction has to be accurate and rapid so that timely medical measures can be taken.

f) Robotics and Autonomous Systems: Path Planning

Path planning has gained importance in the context of robotic and autonomous systems, where a robot or vehicle has to get from an initial point to a final target while determining a path and avoiding obstacles along the way. So, we are assuming an autonomous drone that is tasked with flying through a two-dimensional region where non-moving objects occupy its airspace. The goal, in this case, is to fly the drone with the least travel distance without crashing into any of the stationary barriers.

The optimization variables include:

- Waypoint Coordinates: Set of coordinates of waypoints that lie between this start and end location and will assist the robot to reach the destination.

- Path Smoothness: Guarantees that straight portions of the path do not include any abrupt changes in angles indispensable for real-life applications and effectiveness.

The fitness function for path planning includes:

- Path Length: This is an objective function and a cost that indicates the total distance the robot has traveled, which needs to be as low as possible.
- Obstacle Avoidance: A constraint for the distance to the obstacles, which retains a desired impact of the resulting path that it does not cross any of the obstacles.
- Path Smoothness: A cost function where the aim is limiting the effects of angular displacement and instead maximizing a tendency to create smooth curved paths.

PSO path planning task is also compared to some of the other optimization techniques:

- Genetic Algorithms (GA)
- Ant Colony Optimization (ACO)
- Differential Evolution (DE)
- Simulated Annealing (SA)
- Whale Optimization Algorithm (WOA)

The tables highlight the success rate of the implemented algorithms, and below are the key metrics in which they are executed. Table 17, for instance, captures the number of times the path length has to be optimized for the robot. Table 18 includes a ratio of successful failure-free trials involving the robot, while Table 19 encompasses the time taken for an optimal path to be formulated for the given algorithm.

**Table 17:** Path length comparison

| Algorithm | Path length (meters) |
|---|---|
| Particle Swarm Optimization (PSO) | 35.4 |
| Genetic Algorithms (GA) | 37.2 |
| Ant Colony Optimization (ACO) | 38.0 |
| Differential Evolution (DE) | 36.5 |
| Simulated Annealing (SA) | 39.1 |
| Whale Optimization Algorithm (WOA) | 35.8 |

**Table 18:** Collision-free success rate

| Algorithm | Success rate (%) |
|---|---|
| Particle Swarm Optimization (PSO) | 98 |
| Genetic Algorithms (GA) | 94 |
| Ant Colony Optimization (ACO) | 92 |
| Differential Evolution (DE) | 96 |
| Simulated Annealing (SA) | 90 |
| Whale Optimization Algorithm (WOA) | 97 |

**Table 19:** Computation time comparison

| Algorithm | Computation time (seconds) |
|---|---|
| Particle Swarm Optimization (PSO) | 50 |
| Genetic Algorithms (GA) | 65 |
| Ant Colony Optimization (ACO) | 75 |
| Differential Evolution (DE) | 55 |
| Simulated Annealing (SA) | 80 |
| Whale Optimization Algorithm (WOA) | 60 |

Path Length: Regarding efficiency in route finding, PSO was the best as it completed the route that measured 35.4 m in length. WOA and DE also performed well, achieving path lengths of 35.8 and 36.5 m, respectively. In contrast, the ACO and SA methods did not perform well, as the route lengths predicted were on a longer scale. This points towards PSO dominance in that it retained equilibrium between exploration and exploitation and hence managed to find an optimized route.

Collision-Free Success Rate: PSO ended up being the best with a success rate of 98%, demonstrating its obstacle avoidance ability. The high success rate thus indicates that the waypoint adjustment during the search process of a Modified PSO is reasonably effective in preventing collisions. While ACO and SA had lower rates than the others, both WOA and DE achieved respectable success rates, which confirms PSO to be one of the best in avoiding virtual obstacles and finding paths without collisions.

Computation Time: Out of all the models and techniques used, the PSO was able to complete the route planning in the least time amount of 50 s, which improves its usability and resource deployment for real-time applications. SA and ACO, on the other hand, took the longest and underperformed since, seemingly, the algorithms don't converge as quickly in a multi-constrained complex environment. Although DE and WOA took a fair amount of time to complete, the speed of convergence of PSO, on the other hand, further affirms its positive implications in planning routes in real-time computing. Fig. 6 explores the feature selection process by comparing the accuracy of different algorithms, the number of features selected, and the associated computation times. This analysis provides insight into the trade-offs between performance and time efficiency in data mining tasks.

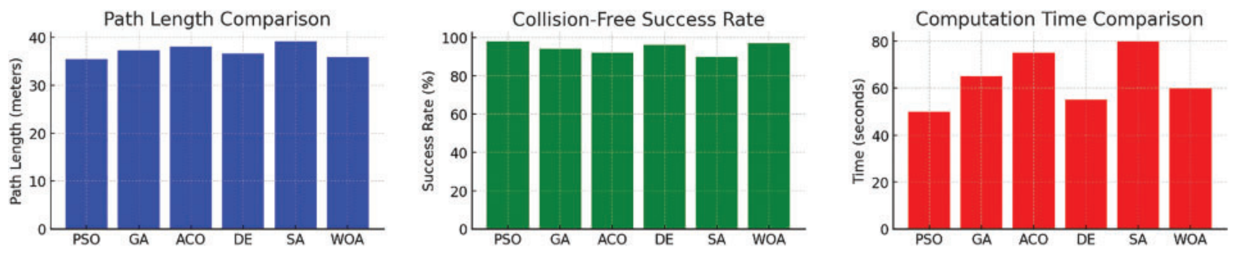

**Figure 6:** The accuracy comparison, number of selected features, and computation time in feature selection

To sum up, it can be said that PSO is superior to GA, ACO, DE, SA, and WOA in obtaining a well-rounded minimum length path plan with the highest percentage ease of solution success with no collision and satisfactory computation times. Add to the fact that PSO can also optimize waypoint selection and path smoothness, it makes PSO the best-suited method for path planning in robotics and autonomous systems, more so if the application deals with real or nearly real-time conditions.

## 6 Future Directions for PSO

Particle Swarm Optimization (PSO) was found to be successful in solving a wide range of optimization problems. However, there are several areas for future development. In addition, hybridization techniques, adaptive mechanisms, and machine learning approaches can be incorporated into PSO to address even more sophisticated challenges in the load forecasting domain. This part considers what improvements might be possible to PSO, what new application areas are opening up, and what trends in optimization tools might guide the development of the PSO.

### *6.1 Improving PSO via Incorporating Hybridization, Adaptive Techniques and Machine Learning*

a) Hybridization Techniques: Despite its challenges, hybridization of PSO is a compelling strategy for performance improvement by integrating it with other optimization methods. In a PSO hybrid model, some limitations include premature convergence, difficulties in constraint handling, and problems with multimodal optimization.

- PSO-hybrid with Evolutionary Algorithms: Some aspects from Genetic Algorithms (GA) or Differential Evolution (DE) can be incorporated with augment the exploration features of PSO and to stop the particles of being confined in the local optima zone. For instance, the particle's position may be improved by incorporating swarm intelligence of PSO and genetic crossover and mutation operators.
- Integration with Local Search Methods: The performance of PSO can be augmented by incorporating local methods like simulated annealing or tabu search, which allows for enhanced exploration and exploitation capabilities. Once the global search is done using PSO, this can be followed by local refinement, which is likely to increase the accuracy of optimal solution centering.
- Multi-Swarm PSO: The deployment of multiple interacting swarms can make the solution robust in the presence of complex landscapes. This allows each swarm to explore different parts of the search space, thereby reducing the chances of becoming trapped in local optimal solutions and enabling better searching.

b) Adaptive Techniques: The introduction of adaptive techniques means that PSO can vary its parameters depending on the nature of the problem, which will often lead to improved performance for that class of problems.

- Dynamic Inertia Weight Adjustment: The strategy inertia weight (ω) must be used to gauge the amount of exploration as compared to exploitativeness appropriately. In contrast to static PSO algorithms, adaptive PSO algorithms initiate with very low values of ω followed by increasing it and subsequently decreasing it when there is increased exploitation.
- Self-Adaptive PSO: A self-adaptive PSO is one in which the values of cognitive and social coefficients (c1 and c2) are altered depending on the work done by the swarm. When the particles are still, θ can be increased to assist in softening the grasp of these coefficients and allow the swarm to disseminate within new areas, hence enhancing performance across different levels of problem difficulties.
- Adaptive Velocity Control: At the particle level, overshooting can occur in very high-dimensional spaces, particularly when the particles are close to the optimum region. This can be controlled by restricting the velocity-sized particles. Employing adaptive velocity control can suitably assist in managing varied levels of the PSO' search space. This is useful *in situ*ations requiring a varying range of solution values.

c) Machine Learning Integration: If PSO integration with machine learning techniques is implemented, the field of application of PSO and its performance in time-varying scenarios will improve.

- Reinforcement Learning for Dynamic Adjustment: The dynamics of the search space can be incorporated into enhancing the particle swarm optimization algorithm by using reinforcement learning in training it. Dynamic optimization, multi-objective optimization, and real-time optimization benefit from reinforcement learning models that can modify and tune the parameters of PSO dynamically.
- Deep Learning-Assisted PSO: PSO can also be used in high-dimensional data, such as neural networks based on deep learning models, for feature selection, hyperparameter optimization, and other complex decision-making tasks. Given that Deep learning assists in informing the swarm on good areas to search in the fitness landscape based on learned trends, there is a good likelihood that Societies employing DL will perform better.
- Transfer Learning in PSO for Knowledge Reuse: TP can aid PSO in transferring knowledge across different problem environments, which would enhance both convergence rate and productivity in related problem areas. This can be particularly useful in scenarios where a similar type of optimization task is performed repeatedly using PSO.

### *6.2 Suggested Research Directions for New Applications of PSO*

a) Optimization in Big Data Analytics: The explosion of big data gives rise to new functions that PSO can be used in, such as data mining, and image recognition, particularly in multivariate data sets. Widened application of modified PSO algorithms that are capable of fitting large data structures is likely to be on the rise in areas like finance, primary care, and marketing, which rely heavily on data.
   - Feature Selection for High-Dimensional Data: It is possible to reduce the number and size of data sets to a PSO that seeks to minimize or classify high-dimensional data and requires lower computation in the domain of machine learning.
   - Parallel and Distributed PSO for Big Data: Parallel and distributed computing PSOs can be effective in handling big data analytics by providing cloud facilities for the intensive calculations needed.

b) Smart Grid and Renewable Energy Optimization: As PSO finds wide acceptance in optimization issues, it should assist in improving the management and integration of renewable energy sources due to the increasing market for sustainable energy.
   - With the deployment of renewable resources and the active participation of consumers, smart grids open up new avenues for PSO methods to shift energy toward peak hours. This enables a number of applications and the use of forecasting, including load forecasting, demand response, and the scheduling of renewable energy.
   - PSO can also help improve the efficiency of the placement of solar panels, wind turbines, and even energy storage batteries in renewable energy systems, which will lower the cost of transitioning to renewable power sources.

c) Bioinformatics and Drug Discovery: PSO's ability to obtain better fitness has great applications in bioinformatics, including optimizing the many processes associated with gene selection, predicting the structure of a protein, and drug design.
   - The class of diseases, including cancer, can be classified at a finer level by using PSO algorithms that lead to the targeting of the more significant genes that were previously missed.
   - Drug optimization includes using PSO for molecular docking. This involves using PSO in drug-finding processes to cohere multiple strands of functionalities into target proteins across candidates.

d) Robotics and Autonomous Systems: The methods can be extended to incorporate path planning or control of PSO for navigation of autonomous vehicles, which can be an interesting area of application.

- Using PSO for path planning can also make balancing navigational safety *vs.* efficiency in complex spaces possible in real time.
- Optimization of Control and Sensor Data Fusion: The need to fine-tune the placement of sensors and control parameters within robotic systems is met through the use of PSO algorithms with improved accuracy in the design of industrial and service robots.

### *6.3 New Areas in Optimization that May Shape the Future Growth of PSO*

a) Multi-Objective and Many-Objective Optimization: The primary focus of earlier versions of PSO was single-objective optimization. However, actual conditions are characterized by multiple conflicting objectives. There is an increasing interest in developing multi-objective PSO (MOPSO) algorithms that deal with objective competition.

- Pareto-Based MOPSO: The use of Pareto-based ranking and selection in MOPSO enables the generation of a set of suboptimal solutions that can satisfy different levels of an objective function. This capability enables the application of MOPSO for engineering design and resource allocation.
- Decomposition-Based MOPSO: In this approach, a multi-objective problem is decomposed into several sub-problems, each optimized by a separate particle swarm. This approach incorporates more diversity in solutions and allows the application to accommodate more complex multi-objectives.

b) Meta-Learning and Hyper-Heuristics: Optimization is now witnessing growing attention to meta-learning, which involves constructing algorithms that are capable of self-improvement. It is possible to include PSO algorithms into hyper-heuristic frameworks to teach them how to optimize for different types of problems, increasing the robustness of the methods.

- Hyper-Heuristic PSO for Optimization Strategy Selection: This combination enables the development of PSO with new search strategies, which switch to the optimum one depending on their effectiveness and improvement—thus making PSO algorithms more general for different problems.
- Tuning of PSO Parameter Using Meta-Learning: The meta-learning method can be applied to tune PSO parameters automatically, making them more suitable for other problems without requiring any manual tuning.

c) Quantum Computing for Optimization: Optimization in quantum computing is improved by utilizing parallelism in a quantum way. Quantum-grown PSO Algorithms based on the angles of Quantum mechanics have the potential to cover the drawbacks of PSO classical efforts.

- Quantum Particle Swarm Optimization (QPSO): QPSO is an updated version of the Particle Swarm Optimization Algorithm. Several concepts in physics, which include quantum superposition, highlight QPSO such that probabilistic position updates of particles are modeled and thus cover a wider range of solutions and even the most optimum one, perhaps so as to prevent the emergence of particles' coherence.
- Hybrid Quantum-Classical Particle Swarm Optimization algorithms: Hybrid Algorithms can be developed by deploying quantum computing over classical PSO, and this will enable high speed and performance to be made possible for the algorithms since the dimensional and complexity of optimization problems will be well catered for.

d) Interpretable Optimization Algorithms: As people understand the relevance of explainability in Artificial Intelligence and machine learning, they desire interpretable optimization models. This Model strives to have simplified PSO representations that will be lucidly understood in the search algorithms. Thus, understanding and robust trust in the optimization outcome are enhanced.

- Showing the PSO Search Process in Motion: These techniques can provide an understanding of particles' motion over generations, enabling practitioners to appreciate how solutions change over time and where they think improvements can be made.
- Explainable PSO for Sensitive Applications: In areas such as health care and finance, the use of this class of explainable PSO strategies may augment trust and, therefore, enhance use.

The future directions outlined here for PSO indicate that the algorithm can progress with the dynamics of improvement, machine learning, and new technologies, making it of great importance for new, more intricate, and more sensitive cases. Moreover, recent papers published many beneficial reviews in the literature for PSO can be found in [75–77].

## 7 Conclusion

The above reminds us that Particle Swarm Optimization (PSO) is a rather complex and efficient optimization algorithm that can be applied in many fields. This paper gives an up-to-date review of PSO by tracing its history, fundamentals, major components, and recent evolution. In contrast with other optimization algorithms and many optimization case studies, this research provides the development, limitations, and future scope of PSO in optimization.

Due to its basic core philosophy of swarm intelligence, PSO can be applied effectively to global functional optimization problems. PSO provides an efficient cooperative search mechanism by allowing the searcher to modify positions according to the actions of the best individual and best global experiences, which speeds up the convergence to optimal or near-optimal solutions. It is because of its ease of use, flexibility, and low operational cost that PSO has become effective in tackling maximum and minimum optimization multivariable problems regardless of their complexity. In various fields including machine learning, engineering design, and energy management, PSO has shown good performance. However, PSO suffers from some limitations, such as premature convergence, strong dependence on parameter settings, and many others, which lead to less performance when optimization problems have relatively high dimensions and are dynamic. These gaps have necessitated improvements and hybridization of PSO methods along with several adaptive strategies.

The analysis that has been conducted within this investigation signifies important aspects of the PSO regarding its competitors, such as Genetic Algorithms, Ant Colony Optimization, and Differential Evolution. Key takeaways include:

- Performance in Continuous Space: For tasks within continuous, higher-dimensional optimization space, PSO appears to be more efficient and usually minimizes within a shorter time frame than other algorithms. However, it does get a disadvantage in multi-dimensional discrete and combinatorial, as ACO may be more adequate here.
- Adaptability and Consistency: A notable advancement in how PSO has been implemented, however, is that it is shown to be more adaptable across a range of optimization tasks. However, there are certain limitations to what PSO is capable of achieving without the aid of hybridization or more advanced adaptive mechanisms in harsher, more dynamic landscapes.
- Need for Parameter Selection: It is observed that the performance of PSO is sensitive to the selected parameters, particularly for the task of convergence time minimization or stability maintenance, which often affect solution situs. Fine-tuning of parameters, such as strategies combined with adaptive ones, improves overall performance, making PSO much more versatile in terms of its applications.

The possibilities for PSO's adaptability and effectiveness are additionally demonstrated by its use cases in data mining, healthcare, and robotics. Particularly promising for practical applications are those where the need for optimization is efficient and scalable. In addition, it has enabled optimization of tasks such as high dimensional feature selection, renewable energy sources management, and robot path planning—which draws attention to its importance in dynamic areas that require complex optimization solutions.

As optimization problems become increasingly widespread in various industries and scientific work, PSO remains an important tool that can be improved further. The improvement of hybridization, adaptive mechanisms, and machine learning incorporation should, on the other hand, increase PSO's relevance and effectiveness in more complex situations. New areas such as bioinformatics and smart grid modeling, along with some new trends in quantum optimization and multi-objective optimization, show new avenues that PSO could pursue for more complex and valuable works.

PSO can be considered an optimization algorithm that has a relatively low span of complexity while still being able to target most if not all, optimization issues. Given these possibilities, it is also inevitable that along with innovations in other industries, the PSO systems will somehow play an important role in future developments in optimizing solutions and making processes more efficient. In applying PSO, its capabilities can be utilized and tailored to the specifics of the performance task, therefore making it an integral part of the optimization toolkit for today's and tomorrow's problems penetration.

**Acknowledgment:** Not applicable.

**Funding Statement:** The author received no specific funding for this study.

**Availability of Data and Materials:** Not applicable.

**Ethics Approval:** Not applicable.

**Conflicts of Interest:** The author declares no conflicts of interest to report regarding the present study.

## References

1. Tien JM. Internet of things, real-time decision making, and artificial intelligence. Ann Data Sci. 2017;4(2):149–78. doi:10.1007/s40745-017-0112-5.
2. Krzywanski J, Sosnowski M, Grabowska K, Zylka A, Lasek L, Kijo-Kleczkowska A. Advanced computational methods for modeling, prediction and optimization—a review. Materials. 2024;17(14):3521. doi:10.3390/ma17143521.
3. Zhang W, Gu X, Tang L, Yin Y, Liu D, Zhang Y. Application of machine learning, deep learning and optimization algorithms in geoengineering and geoscience: comprehensive review and future challenge. Gondwana Res. 2022;109:1–17. doi:10.1016/j.gr.2022.03.015.
4. Antunes CH, Henriques CO. Multi-objective optimization and multi-criteria analysis models and methods for problems in the energy sector. Multi Criteria Deci Anal: State Art Surv. 2016;233:1067–165.
5. Osaba E, Villar-Rodriguez E, Del Ser J, Nebro AJ, Molina D, LaTorre A, et al. A tutorial on the design, experimentation and application of metaheuristic algorithms to real-world optimization problems. Swarm Evol Comput. 2021;64:100888. doi:10.1016/j.swevo.2021.100888.
6. Sarker IH. Data science and analytics: an overview from data-driven smart computing, decision-making and applications perspective. SN Comput Sci. 2021;2(5):377. doi:10.1007/s42979-021-00765-8.
7. Efendigil T, Önüt S, Kahraman C. A decision support system for demand forecasting with artificial neural networks and neuro-fuzzy models: a comparative analysis. Expert Syst Appl. 2009;36(3):6697–707. doi:10.1016/j.eswa.2008.08.058.
8. Kaveh M, Mesgari MS. Application of meta-heuristic algorithms for training neural networks and deep learning architectures: a comprehensive review. Neural Process Lett. 2023;55(4):4519–622. doi:10.1007/s11063-022-11055-6.
9. Blocho M. Heuristics, metaheuristics, and hyperheuristics for rich vehicle routing problems. In: Smart delivery systems. Intelligent Data-Centric Systems; 2020. p. 101–56.

10. Hussain K, Mohd Salleh MN, Cheng S, Shi Y. Metaheuristic research: a comprehensive survey. Artif Intell Rev. 2019;52(4):2191–233. doi:10.1007/s10462-017-9605-z.
11. Abualigah L, Sheikhan A, Ikotun AM, Zitar R, Alsoud A, Al-Shourbaji I, et al. Particle swarm optimization algorithm: review and applications. Metaheur Optimiz Algor. 2024:1–14. doi:10.1016/B978-0-443-13925-3.00019-4.
12. Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of ICNN'95-International Conference on Neural Networks; 1995; Perth, WA, Australia. p. 1942–8. doi:10.1109/ICNN.1995.488968
13. Shami TM, El-Saleh AA, Alswaitti M, Al-Tashi Q, Summakieh MA, Mirjalili S. Particle swarm optimization: a comprehensive survey. IEEE Access. 2022;10:10031–61. doi:10.1109/ACCESS.2022.3142859.
14. Marini F, Walczak B. Particle swarm optimization (PSO). A tutorial. Chemometr Intell Lab Syst. 2015;149:153–65. doi:10.1016/j.chemolab.2015.08.020.
15. Holland JH. Genetic algorithms. Sci Am. 1992 Jul;267(1):66–73.
16. Yao X. A new simulated annealing algorithm. Int J Comput Mathem. 1995;56(3–4):161–8. doi:10.1080/00207169508804397.
17. Du KL, Swamy MNS. Harmony search. In: Search and optimization by metaheuristics. Cham: Springer International Publishing; 2016. p. 227–35.
18. Zhang Y, Wang S, Ji G. A comprehensive survey on particle swarm optimization algorithm and its applications. Math Probl Eng. 2015;2015(1):931256. doi:10.1155/2015/931256.
19. Tang J, Liu G, Pan Q. A review on representative swarm intelligence algorithms for solving optimization problems: applications and trends. IEEE/CAA J Automat Sinica. 2021;8(10):1627–43. doi:10.1109/JAS.2021.1004129.
20. Zaini FA, Sulaima MF, Razak IAWA, Zulkafli NI, Mokhlis H. A review on the applications of PSO-based algorithm in demand side management: challenges and opportunities. IEEE Access. 2023;11:53373–400. doi:10.1109/ACCESS.2023.3278261.
21. Qaraad M, Amjad S, Hussein NK, Farag MA, Mirjalili S, Elhosseini MA. Quadratic interpolation and a new local search approach to improve particle swarm optimization: solar photovoltaic parameter estimation. Expert Syst Appl. 2024;236:121417. doi:10.1016/j.eswa.2023.121417.
22. Thaher T, Chantar H, Too J, Mafarja M, Turabieh H, Houssein EH. Boolean particle swarm optimization with various evolutionary population dynamics approaches for feature selection problems. Expert Syst Appl. 2022;195:116550. doi:10.1016/j.eswa.2022.116550.
23. Houssein EH, Sayed A. Boosted federated learning based on improved Particle Swarm Optimization for healthcare IoT devices. Comput Biol Med. 2023;163:107195. doi:10.1016/j.compbiomed.2023.107195.
24. Yazdanjue N, Yazdanjouei H, Karimianghadim R, Gandomi AH. An enhanced discrete particle swarm optimization for structural k-Anonymity in social networks. Inf Sci. 2024;670:120631. doi:10.1016/j.ins.2024.120631.
25. Hu G, Guo Y, Zhao W, Houssein EH. An adaptive snow ablation-inspired particle swarm optimization with its application in geometric optimization. Artif Intell Rev. 2024;57(12):332. doi:10.1007/s10462-024-10946-5.
26. Nishat MM, Sakib S, Rahman KA, Ahmed M, Shagor Md RK, Faisal F, et al. Applying particle swarm optimization for investigating the stability of DC-DC buck-boost converter: a software based approach. Int J Model Simul. 2024:1–20. doi:10.1080/02286203.2024.2315324.
27. Piotrowski AP, Napiorkowski JJ, Piotrowska AE. Particle swarm optimization or differential evolution—a comparison. Eng Appl Artif Intell. 2023;121:106008. doi:10.1016/j.engappai.2023.106008.
28. Jahandideh-Tehrani M, Bozorg-Haddad O, Loáiciga HA. Application of particle swarm optimization to water management: an introduction and overview. Environ Monit Assess. 2020;192:281. doi:10.1007/s10661-020-8228-z.
29. Piotrowski AP, Napiorkowski JJ, Piotrowska AE. Population size in particle swarm optimization. Swarm Evol Comput. 2020;58:100718. doi:10.1016/j.swevo.2020.100718.
30. Wang F, Zhang H, Zhou A. A particle swarm optimization algorithm for mixed-variable optimization problems. Swarm Evol Comput. 2021;60:100808. doi:10.1016/j.swevo.2020.100808.
31. Zeng N, Wang Z, Liu W, Zhang H, Hone K, Liu X. A dynamic neighborhood-based switching particle swarm optimization algorithm. IEEE Trans Cybern. 2022;52(9):9290–301. doi:10.1109/TCYB.2020.3029748.

32. Pozna C, Precup R-E, Horvath E, Petriu EM. Hybrid particle filter-particle swarm optimization algorithm and application to fuzzy controlled servo systems. IEEE Trans Fuzzy Syst. 2022;30(10):4286–97. doi:10.1109/TFUZZ.2022.3146986.
33. Zhang X, Liu H, Tu L. A modified particle swarm optimization for multimodal multi-objective optimization. Eng Appl Artif Intell. 2020;95:103905. doi:10.1016/j.engappai.2020.103905.
34. Pace F, Santilano A, Godio A. A review of geophysical modeling based on particle swarm optimization. Surv Geophys. 2021;42(3):505–49. doi:10.1007/s10712-021-09638-4.
35. Wang F, Wang X, Sun S. A reinforcement learning level-based particle swarm optimization algorithm for large-scale optimization. Inf Sci. 2022;602:298–312. doi:10.1016/j.ins.2022.04.053.
36. Nama S, Saha AK, Chakraborty S, Gandomi AH, Abualigah L. Boosting particle swarm optimization by backtracking search algorithm for optimization problems. Swarm Evol Comput. 2023;79:101304. doi:10.1016/j.swevo.2023.101304.
37. Xu Y, Hu C, Wu Q, Jian S, Li Z, Chen Y, et al. Research on particle swarm optimization in LSTM neural networks for rainfall-runoff simulation. J Hydrol. 2022;608(1):127553. doi:10.1016/j.jhydrol.2022.127553.
38. Yang X, Li HJIS. Evolutionary-state-driven multi-swarm cooperation particle swarm optimization for complex optimization problem. Inf Sci. 2023;646(9):119302. doi:10.1016/j.ins.2023.119302.
39. Sedighizadeh D, Masehian E, Sedighizadeh M, Akbaripour H. GEPSO: a new generalized particle swarm optimization algorithm. Math Comput Simul. 2021;179(2):194–212. doi:10.1016/j.matcom.2020.08.013.
40. Luo X, Yuan Y, Chen S, Zeng N, Wang Z. Position-transitional particle swarm optimization-incorporated latent factor analysis. IEEE Trans Knowl Data Eng. 2022;34(8):3958–70. doi:10.1109/TKDE.2020.3033324.
41. Otair M, Ibrahim OT, Abualigah L, Altalhi M, Sumari P. An enhanced grey wolf optimizer based particle swarm optimizer for intrusion detection system in wireless sensor networks. Wireless Netw. 2022;28(2):721–44. doi:10.1007/s11276-021-02866-x.
42. Liu Y, Liu J, Jin Y. Surrogate-assisted multipopulation particle swarm optimizer for high-dimensional expensive optimization. IEEE Transact Syst Man Cybernet: Syst. 2021;52(7):4671–84. doi:10.1109/TSMC.2021.3102298.
43. Song XF, Zhang Y, Gong DW, Sun XY. Feature selection using bare-bones particle swarm optimization with mutual information. Pattern Recognit. 2021;112(4):107804. doi:10.1016/j.patcog.2020.107804.
44. Song X-F, Zhang Y, Gong D-W, Gao X-Z. A fast hybrid feature selection based on correlation-guided clustering and particle swarm optimization for high-dimensional data. IEEE Trans Cybern. 2021;52(9):9573–86. doi:10.1109/TCYB.2021.3061152.
45. Pirozmand P, Jalalinejad H, Hosseinabadi AR, Mirkamali S, Li Y. An improved particle swarm optimization algorithm for task scheduling in cloud computing. J Ambient Intell Humaniz Comput. 2023;14(4):4313–27. doi:10.1007/s12652-023-04541-9.
46. Singh N, Singh S, Houssein EH. Hybridizing salp swarm algorithm with particle swarm optimization algorithm for recent optimization functions. Evol Intel. 2022;15(1):23–56. doi:10.1007/s12065-020-00486-6.
47. Shams MY, El-kenawy EM, Ibrahim A, Elshewey AM. A hybrid dipper throated optimization algorithm and particle swarm optimization (DTPSO) model for hepatocellular carcinoma (HCC) prediction. Biomed Signal Process Contr. 2023;85:104908. doi:10.1016/j.bspc.2023.104908.
48. Jiang JJ, Wei WX, Shao WL, Liang YF, Qu YY. Research on large-scale bi-level particle swarm optimization algorithm. IEEE Access. 2021;9:56364–75. doi:10.1109/ACCESS.2021.3072199.
49. Tian J, Hou M, Bian H, Li J. Variable surrogate model-based particle swarm optimization for high-dimensional expensive problems. Complex Intell Syst. 2023;9(4):3887–935. doi:10.1007/s40747-022-00910-7.
50. Wang R, Hao K, Chen L, Wang T, Jiang C. A novel hybrid particle swarm optimization using adaptive strategy. Inf Sci. 2021;579(6):231–50. doi:10.1016/j.ins.2021.07.093.
51. Tijjani S, Ab Wahab MN, Mohd Noor MH. An enhanced particle swarm optimization with position update for optimal feature selection. Expert Syst Appl. 2024;247(7):123337. doi:10.1016/j.eswa.2024.123337.
52. Li S, Chi X, Yu B. An improved particle swarm optimization algorithm for the reliability-redundancy allocation problem with global reliability. Reliab Eng Syst Saf. 2022;225(2):108604. doi:10.1016/j.ress.2022.108604.

53. Afroz Z, Shafiullah GM, Urmee T, Shoeb MA, Higgins G. Predictive modelling and optimization of HVAC systems using neural network and particle swarm optimization algorithm. Build Environ. 2022;209(6):108681. doi:10.1016/j.buildenv.2021.108681.

54. Jena L, Mishra S, Nayak S, Ranjan P, Mishra M. Variable optimization in cervical cancer data using particle swarm optimization. In: Advances in electronics, communication and computing. Singapore: Springer Nature Singapore; 2021. p. 147–53.

55. Li Y, Chu X, Tian D, Feng J, Mu W. Customer segmentation using K-means clustering and the adaptive particle swarm optimization algorithm. Appl Soft Comput. 2021;113(1):107924. doi:10.1016/j.asoc.2021.107924.

56. Hu Y, Zhang Y, Gong D. Multiobjective particle swarm optimization for feature selection with fuzzy cost. IEEE Trans Cybern. 2020;51(2):874–88. doi:10.1109/TCYB.2020.3015756.

57. Xu L, Song B, Cao M. An improved particle swarm optimization algorithm with adaptive weighted delay velocity. Syst Sci Control Eng. 2021;9(1):188–97. doi:10.1080/21642583.2021.1891153.

58. Kan X, Fan Y, Fang Z, Cao L, Xiong NN, Yang D, et al. A novel IoT network intrusion detection approach based on adaptive particle swarm optimization convolutional neural network. Inf Sci. 2021;568(5):147–62. doi:10.1016/j.ins.2021.03.060.

59. Rashno A, Shafipour M, Fadaei S. Particle ranking: an efficient method for multi-objective particle swarm optimization feature selection. Knowl Based Syst. 2022;245(1):108640. doi:10.1016/j.knosys.2022.108640.

60. Pesaran HAM, Nazari-Heris M, Mohammadi-Ivatloo B, Seyedi H. A hybrid genetic particle swarm optimization for distributed generation allocation in power distribution networks. Energy. 2020;209(12):118218. doi:10.1016/j.energy.2020.118218.

61. Adnan RM, Mostafa RR, Kisi O, Yaseen ZM, Shahid S, Zounemat-Kermani M. Improving streamflow prediction using a new hybrid ELM model combined with hybrid particle swarm optimization and grey wolf optimization. Knowl Based Syst. 2021;230(6):107379. doi:10.1016/j.knosys.2021.107379.

62. Hu Y, Zhang Y, Gao X, Gong D, Song X, Guo Y, et al. A federated feature selection algorithm based on particle swarm optimization under privacy protection. Knowl Based Syst. 2023;260(3):110122. doi:10.1016/j.knosys.2022.110122.

63. Zhang Y, Yuan LJ, Zhang Q, Sun XY. Multi-objective optimization of building energy performance using a particle swarm optimizer with less control parameters. J Build Eng. 2020;32(11):101505. doi:10.1016/j.jobe.2020.101505.

64. Mozaffari S, Javadi S, Moghaddam HK, Randhir TO. Forecasting groundwater levels using a hybrid of support vector regression and particle swarm optimization. Water Resour Manag. 2022;36(6):1955–72. doi:10.1007/s11269-022-03118-z.

65. Qin C, Zhang Y, Bao F, Zhang C, Liu P, Liu P. XGBoost optimized by adaptive particle swarm optimization for credit scoring. Math Probl Eng. 2021;2021(5):6655510. doi:10.1155/2021/6655510.

66. Wu M, Du P, Jiang M, Goh HH, Zhu H, Zhang D, et al. An integrated energy system optimization strategy based on particle swarm optimization algorithm. Energy Rep. 2022;8(12):679–91. doi:10.1016/j.egyr.2022.10.034.

67. Jin X, Wei B, Deng L, Yang S, Zheng J, Wang F. An adaptive pyramid PSO for high-dimensional feature selection. Expert Syst Appl. 2024;257(12):125084. doi:10.1016/j.eswa.2024.125084.

68. Hu H, Fan X, Wang C. Energy efficient clustering and routing protocol based on quantum particle swarm optimization and fuzzy logic for wireless sensor networks. Sci Rep. 2024;14(1):18595. doi:10.1038/s41598-024-69360-0.

69. Alhijawi B, Awajan A. Genetic algorithms: theory, genetic operators, solutions, and applications. Evol Intell. 2024;17(3):1245–56. doi:10.1007/s12065-023-00822-6.

70. Liu C, Wu L, Li G, Xiao W, Tan L, Xu D, et al. AI-based 3D pipe automation layout with enhanced ant colony optimization algorithm. Autom Constr. 2024;167:105689. doi:10.1016/j.autcon.2024.105689.

71. Wang J, Wang Y. An efficient improved whale optimization algorithm for optimization tasks. Eng Letters. 2024;32(2):392.

72. Wang H, Liang Q, Hancock JT, Khoshgoftaar TM. Feature selection strategies: a comparative analysis of SHAP-value and importance-based methods. J Big Data. 2024;11:44. doi:10.1186/s40537-024-00905-w.

73. Sadeghian O, Shotorbani AM, Ghassemzadeh S, Mohammadi-Ivatloo B. Energy management of hybrid fuel cell and renewable energy based systems—a review. Int J Hydrogen Energy. 2024. doi:10.1016/j.ijhydene.2024.03.134.
74. Reda M, Onsy A, Haikal AY, Ghanbari A. Path planning algorithms in the autonomous driving system: a comprehensive review. Robot Auton Syst. 2024;174:104630. doi:10.1016/j.robot.2024.104630.
75. Nayak J, Swapnarekha H, Naik B, Dhiman G, Vimal S. 25 years of particle swarm optimization: flourishing voyage of two decades. Arch Comput Methods Eng. 2023;30(3):1663–725. doi:10.1007/s11831-022-09849-x.
76. Kuo R, Li S-S. Applying particle swarm optimization algorithm-based collaborative filtering recommender system considering rating and review. Appl Soft Comput. 2023;135:110038. doi:10.1016/j.asoc.2023.110038.
77. Wakchaure M, Patle B, Mahindrakar A. Application of AI techniques and robotics in agriculture: a review. Artif Intell Life Sci. 2023;3:100057. doi:10.1016/j.ailsci.2023.100057.