Comparative Analysis of Sports-Inspired Optimization Algorithms in Neural Network Training: A Benchmark Study

Eyad J. R. Elfarra[1], Mohammed A. Alhanjouri[2]

Computer Engineering Department, Collage of Engineering, The Islamic university of Gaza, Gaza, Palestine.

Email: [1]efarra@iugaza.edu.ps, [2]mhanjouri@iugaza.edu.ps

Abstract

Nine population-based optimization algorithms, six of which are sports-inspired and three classical natural-inspired, are applied in this work to neural network hyperparameter tuning. The results of the sport-inspired algorithms, namely the Most Valuable Player Algorithm, Soccer League Champions Algorithm, Golden Ball Algorithm, Tiki-Taka Algorithm, Tug of War Optimization, and World Cup Optimization, will be compared against classical metaheuristics, namely Particle Swarm Optimization, Differential Evolution, and Genetic Algorithm. Each optimizer was evaluated under identical experimental conditions on a neural network classifier using the Iris dataset to optimize four key hyperparameters: the number of hidden layers, neurons per layer, learning rate, and L2 regularization coefficient. Due to its hierarchical competition structure, MVPA had the highest classification accuracy (99.33%), followed by SLCA (98.67%), but it came at a higher computational cost. GBA and PSO showed very strong performances with efficient convergence, while DE, TOW, and WCO resulted in relatively lower accuracies of ~97–98% but achieved fast convergence, which is very suitable for time-critical applications. The results show that sports-inspired metaheuristics are robust and effective frameworks for neural network hyperparameter optimization. Their competition-driven strategic mechanisms enable efficient search behavior that is on par with, and occasionally even better than, that of more conventional evolutionary algorithms. These results provide a solid basis for hybrid, adaptive algorithms in the future related to sports.

Keywords: Sport-Inspired Optimization, Hyperparameter Optimization, Deep Learning Optimization, Population-based, Algorithms.

DOI: https://doi.org/10.36548/jaicn.2025.4.004
Received: 10.10.2025, received in revised form: 11.11.2025, accepted: 26.11.2025, published: 05.12.2025

1. Introduction

The optimal performance of the neural network (NN) is achieved by tuning its hyperparameters. The hyperparameters can be categorized into two categories: architecture and learning rate dynamics hyperparameters [1]. The architecture parameters, such as the number of hidden layers and the size of each hidden layer, are discrete values, while the learning rate dynamics, such as the learning rate and regularization coefficient (α), are continuous values that need to be optimized. The search space for these hyperparameters is high-dimensional, nonlinear, and frequently filled with multiple local optima [2]. Traditional approaches like grid search or random search exhibit exponential scaling and limited adaptability, and gradient-based methods (e.g., Bayesian optimization variants) sometimes struggle when the parameter space is discrete/mixed or highly non-smooth.[3].

To efficiently navigate this complex space, Population-Based Optimization Algorithms (POAs) which are a category of metaheuristics that simulate natural phenomena like biology and sports have been widely used for hyperparameter optimization (HPO).[4] These methods trade off exhaustive search for guided exploration and exploitation, enabling them to probe large, irregular parameter spaces more efficiently.[5] Sports-inspired optimization algorithms (SIOAs) have gained interest because they draw from competitive, strategic, or cooperative dynamics observed in sports, and they often embody teamwork, contest, substitution, ranking, and scheduling dynamics, thus often giving rise to unique mechanisms.[6]

Making a reliable advancement in optimization techniques demands benchmarks that overcome three experimental challenges: accurately measuring the total computational effort required to achieve the desired result, accounting for the sensitivity of measurements to specific workload details, and ensuring a fair comparison among algorithms, all of which often require complex internal hyperparameter tuning themselves [16]. To address this, the current study utilizes standardized, fixed-budget evaluation protocols and statistical significance testing, ensuring that the comparison of efficiency and robustness is fair, objective, and reproducible.

This study makes key contributions to computational intelligence and deep learning optimization by presenting a comprehensive benchmark comparing nine population-based optimization algorithms on neural network hyperparameter optimization. The three classical metaheuristics algorithms are: Particle Swarm Optimization (PSO) [9], Genetic Algorithm (GA) [7], and Differential Evolution (DE) [8]. The six Sports-Inspired Optimization

Algorithms (SIOAs) used here are Golden Ball (GB) [10], Most Valuable Player Algorithm (MVPA) [11], Soccer League Championship Algorithm (SLCA) [12], Tiki-Taka Algorithm (TTA) [13], Tug-of-war Optimization (TWO)[14], and World Cup Optimization (WCO)[15]

This research evaluates whether sports-inspired search strategies, such as short-passing in TTA or team transfers in GB, improve efficiency, optimality, and robustness over traditional evolutionary and swarm methods. The research further analyses algorithm performance on heterogeneous search spaces with both discrete and continuous parameters and introduces standardized robustness and efficiency metrics, establishing new empirically validated baselines for hyperparameter optimization.

2. Related work

Interest in automating hyperparameter selection has continued to grow, with several recent reviews and benchmarks that synthesize classical sequential model-based approaches (e.g., Bayesian optimization) alongside population-based and metaheuristic methods. Comprehensive surveys highlight the continuing relevance of black-box, population-based optimizers for mixed and discrete search spaces where gradient or surrogate methods may struggle, and they provide unified benchmarking frameworks for comparing methods at scale. Recent significant reviews consist of a 2024 systematic analysis of HPO techniques that highlight where population/metaheuristic approaches remain competitive [22].

Classical population-based optimizers are still commonly employed for HPO in neural networks because of their adaptability to work with discrete and mixed parameter types, along with their implementation simplicity. Recent research and case studies show DE variants successfully designing CNN architectures and optimizing the training hyperparameters for domain-specific tasks [8]. El Hassani et al. (2024) proposed a real-coded genetic algorithm for optimizing MLP hyperparameters, improving network performance. This approach efficiently balances exploration and exploitation in complex search spaces [23]. Salem et al. (2024) applied Particle Swarm Optimization to tune hyperparameters of ML models for COVID-19 big data, enhancing prediction accuracy. Their approach improved performance across Random Forests, SVMs, Decision Trees, and Neural Networks [24].

Sports-inspired optimization algorithms (SIOAs) model competitive and cooperative phenomena from sports (tournaments, leagues, passing/tactics) to create novel search operators. While many original SIOA proposals predate 2020, there has been a steady stream of recent work applying, extending, or hybridizing these algorithms for engineering and ML tasks. Golden Ball & League/World Cup variants — Golden Ball and league/tournament based methods (including World Cup Optimization) have been adapted and enhanced in recent years (e.g., learning-enhanced Golden Ball, 2022) and applied in optimization tasks such as routing and classification problems; the World Cup algorithm has been used in medical image and melanoma detection tasks and continues to appear in domain applications[10].Tug-of-War Optimization (TWO), while introduced earlier, has continued to be referenced and hybridized in structural and engineering optimization work post-2020; reviews of TWO emphasize its unique force-balance update mechanism that can improve global search in some continuous domains.[14].

Although sports-inspired algorithms are increasingly applied to engineering problems, systematic evaluations of multiple SIOAs specifically for neural network hyperparameter optimization are limited. The existing SIOA literature is rich in benchmark function comparisons and domain case studies but rarely unifies many SIOAs under consistent HPO experimental setups for neural models; this gap motivates the present benchmark study [13].

From the recent literature (2020–2024), three consistent gaps were identified that motivate this study:

1. Comparative SIOA evaluations for HPO: Multiple sports-inspired algorithms exist and have been applied to optimization tasks, but there is a lack of unified benchmarking that compares several SIOAs and classical metaheuristics for neural network hyperparameter tuning under the same experimental framework [13].
2. Empirical HPO guidance for mixed/discrete hyperparameters: While Bayesian and surrogate approaches dominate HPO surveys, less attention has been given to how SIOAs behave when the search space mixes discrete structural choices (Num_Layers, Layer_Size) and continuous hyperparameters (Learning_Rate, Alpha). Recent surveys highlight this as a practical shortcoming.[22]
3. Computational cost vs. performance trade-offs: Recent DE/GA/PSO works and DE-HPO studies emphasize efficiency improvements, but few studies quantify the runtime and evaluation budget tradeoffs of sports-inspired algorithms specifically for neural model HPO.

These gaps motivate this benchmark to implement and evaluate DE, GA, and PSO (classical) alongside GoldenBall, MVPA, SLCA, Tiki-Taka, TugOfWar, and WorldCup (sports-inspired) on a common HPO task optimizing Num_Layers, Layer_Size, Learning_Rate, and Alpha across multiple datasets and budgets to quantify relative performance, convergence, and computational cost.

3. Neural Network Hyperparameter Search Space Characterization

Neural network optimization consists of two primary, interconnected types of hyperparameters: those that regulate the architectural design (which determines the model capacity), and those that control the optimization dynamics (which govern the training trajectory and speed).[4] Simultaneous optimization across these distinct categories creates a highly challenging, interdependent search landscape.

Stochasticity plays a pivotal role in balancing the depth of the hidden layer and the distribution of neurons. Random distributions allow candidate solutions to explore alternative structures that might not be reached by deterministic updates. This stochasticity helps prevent early convergence toward narrow structures and encourages the exploration of deeper or broader configurations when needed.

The four hyperparameters selected for this benchmark intentionally span both structural and rate dimensions, demanding robust performance from the metaheuristic optimizers in a complex, mixed-integer search space.

3.1 Structural Hyperparameters (Discrete Search Space)

Structural hyperparameters define the topology and capacity of the deep learning model. They are typically integer-valued, introducing constraints that require metaheuristics to integrate effective strategies for handling discrete search components.

3.1.1 Number of Layers

This parameter specifies the network's depth. Many researches indicate that increasing a network's depth often improves generalization and test accuracy compared to simply increasing layer width, though greater depth also compounds training difficulty.[17] Since this is a discrete, integer-valued parameter, algorithms defined primarily for continuous spaces must employ specific rounding, discretization, or specialized integer operators for mutation and crossover.[18]

3.1.2 Hidden Layers Size

This parameter determines the width of each layer, specifically the number of units or neurons per layer.[19] The layer size profoundly affects the model's representational capacity and its overall computational complexity. The necessity for metaheuristics to identify optimal values within Optimization Rate Hyperparameters (Continuous Search Space)

3.2 The Weight Update and Generalization Hyperparameters

3.2.1 Learning Rate

The learning rate parameter determines the step size taken in the direction of the loss function's negative gradient. Metaphorically, it represents the speed at which the model acquires and incorporates new information ("learns").[20] The selection of this rate must be done carefully, because a high learning rate can lead to instability and divergence by overshooting the minima, while a low learning rate may result in extremely slow convergence or falling into an undesirable local minimum.[20] The optimal learning rate is typically sought within a broad, often logarithmic, continuous range (e.g., 10-4 to 10-1).

3.2.2 L2 Regularization (α)

The α parameter in L2 regularization controls the penalty on large model weights to prevent overfitting. By adding an α term to the loss function, it balances bias and variance larger α increases regularization, enhancing generalization but possibly reducing accuracy [21]. L2 regularization interacts with learning rate and model capacity. This makes it an important hyperparameter that affects generalization. Treating α as a parameter to optimize helps find configurations that best balance bias and variance for each architecture. Optimizing network topology is inherently more complex than tuning scalar parameters such as learning rate. The depth and width of the NN define a combinatorial space with nonlinear interactions affecting capacity, expressiveness, and generalization, while the learning rate occupies a smooth, continuous one-dimensional interval.

4. Theoretical Foundations of Baseline Population-Based Algorithms

First, to establish a quantitative standard for assessing the SIOAs, a review of the foundational algorithms is needed, focusing on the basic inherent strategies adopted therein to balance the trade-off between exploration-diversification of search-and exploitation-intensification of search in promising regions.

4.1 Genetic Algorithm (GA)

Genetic Algorithm is an evolutionary optimization algorithm inspired by natural selection, which improves iteratively a set of candidate solutions (chromosomes) [7]. Its effectiveness depends on exploitation-achieved through selection and crossover, where the fittest individuals are chosen as parents and their genetic material combined to produce better offspring while preserving elite solutions-and exploration driven by mutation, which introduces small random variations with the purpose of maintaining diversity and helping to avoid local optima, thus ensuring that a wider region of the solution space is searched [7].

4.2 Differential Evolution (DE)

DE is one of the most powerful POA algorithms that works very effectively on complex, nonlinear, and nonconvex problems. It implements the evolution of a set of candidate solutions using only basic vector operations and does not require gradient information, which is appropriate for such discontinuous environments as hyperparameter optimization. For each agent, DE generates a mutated vector based on the differences between three random agents and then combines the mutated vector with the target vector through crossover before selection. This mechanism can efficiently balance exploration and exploitation.

4.3 Particle Swarm Optimization (PSO)

PSO is a swarm intelligence optimization technique where every solution is represented by a particle characterized by its position, velocity, and fitness value. In this context, each particle moves with a velocity that is determined by its personal best position, known as (pBest), and the global best position found by the swarm. Three main coefficients contribute to the balance between exploration and exploitation: inertia weight for momentum control, the cognitive coefficient for individual learning, and the social coefficient for collective learning and cooperation.

5. Sports-Inspired Optimization Algorithms (SIOAs)

The six SIOAs under investigation utilize specific, complex sports metaphors to define their search operators. A critical element of this benchmark is determining whether these complex metaphorical mechanisms yield a functional advantage over the simplified and well-established search strategies of GA, DE, and PSO, or whether they merely introduce semantic novelty to existing evolutionary mechanics.

Dynamic competitive behaviors like player transfers, role switching, match simulations, short-pass strategies, and hierarchical leagues are all incorporated into sports-inspired algorithms. Both local exploitation and global exploration are supported by these behaviors, which alter the candidate solutions according to comparative fitness. These systems aid in preserving diversity and avoiding stagnation.

5.1 Golden Ball (GB) Optimization Algorithm

The Golden Ball (GB) meta-heuristic is an optimization algorithm inspired by soccer that utilizes a multi-population structure organized into teams within a league [10]. Exploitation (intensification) takes place in training phases, where the team's captain and coach employ tailored crossover operations to enhance the best performing solutions [10]. Exploration (diversification) is encouraged via several autonomous teams and a transfer period that facilitate player (solution) exchanges among teams based on league rankings, improving global search and avoiding early convergence [10].

5.2 Tiki-Taka Algorithm (TTA)

The Tiki-Taka Algorithm (TTA) is inspired by Tiki-Taka, a football strategy characterized by short passing and continuous movement [13]. Localized short passing (a player passes the "ball" to the nearest-neighbor player in order to move to a superior position) enables exploitation within TTA [13]. Exploration is maintained by deriving the movement of players with respect to multiple key player positions instead of one single global best, preserving diversity while guiding the search via a balanced, neighborhood-based approach [13].

5.3 Tug of War Optimization (TWO)

The TWO algorithm represents potential solutions as teams engaged in a rope-pulling contest [14]. Search is guided by attractive forces proportional to solution quality, steering solutions toward superior areas [14]. The competitive interplay of the algorithm between exploitation and exploration is balanced, wherein strong solutions guide the search while weaker solutions maintain diversity [14].

5.4 World Cup Optimization (WCO)

The WCO takes its inspiration from the hierarchical structure of the FIFA World Cup [15]. The search is conducted in stages, moving through preliminaries to group rounds and play-offs across populations segmented into groups, or continents [15]. Exploration is achieved by promoting high fitness individuals through the stages in an iterative refinement of elite solutions. Also similar to the team-based approach of the Golden Ball algorithm, the group-based structure maintains diversity and avoids premature convergence [15].

5.5 Soccer League Championship Algorithm (SLCA)

The Soccer League Championship Algorithm represents a competitive sports league and models it as teams of candidate solutions whose strength reflects fitness [12]. It guides the search by simulated match outcomes, which adjust team formations (solution vectors) iteratively. Exploration enhancement is achieved by mechanisms such as generating multiple offspring from successful solutions and allowing high-quality infeasible solutions to survive, which enhances diversity and expands the global search [12].

5.6 Most Valuable Player Algorithm (MVPA)

The MVPA is inspired by the concept of leveraging the "Most Valuable Player" and is designed for fast, efficient, and reliable performances in time-consuming engineering optimization problems [11]. Clearly, the focus of MVPA is on exploitation to converge quickly to good solutions. However, such a speed-oriented strategy may not be able to locate the global optimum in difficult, multimodal landscapes and may converge quickly to local optima. Indeed, convergence speed and solution quality must be properly balanced while benchmarking MVPA against more explorative algorithms like DE or GB [11].

6. Methodology

This work presents a comparative benchmark analysis of nine optimization algorithms to tune the hyperparameters of the neural network classifier, of which six are sports-inspired optimization algorithms.

In this paper, the focus is on hyperparameter optimization for MLP, taken as a benchmark on the Iris dataset. MLP algorithms have been chosen to isolate optimizer behavior from structural complexity. CNNs and deep structures introduce domain-specific constraints and significantly higher computational costs, which are likely to push the standard towards algorithms that may exploit early stopping or parameter sharing. In such contexts, simplified MLP allows for an apples-to-apples comparison of the optimizer mechanisms. The methodology integrates FE, population-based optimization strategies, and systematic evaluation through cross-validation. All the algorithms were implemented under a uniform experimental framework in Python, with their main objective being the minimization of the classifier validation error. Code is run in Google Collab.

This section describes the structured approach followed in this work, including preparing the dataset, designing the models, optimizing hyperparameters, and testing the experimental settings. Figure 1 presents the workflow that has been followed during the research.

6.1 Dataset Selection

Experiments were conducted using the Iris dataset from the UCI Machine Learning Repository. This dataset is made up of 150 samples in three classes-setosa, versicolor, and virginica-each described by four numeric features: sepal length, sepal width, petal length, and petal width.

6.2 Data Pre-processing

All the features are normalized into the range of [0,1] beforehand as a pre-processing step in order to ensure that any difference in performances arises from the different optimization strategies and not data inconsistencies.

A stratified random split was performed on the dataset, which preserved the class balance across subsets, 80% training and 20% validation.

6.3 Hyperparameter Definition and Bounds

All experiments used the neural network "MLPClassifier" from "sklearn.neural_network". Its activation function was fixed to "ReLU". Bounds ensure feasible and efficient coverage of the search space. More precisely, multiple structures and hyperparameters were dynamically defined according to the output vector of each optimizer. The optimization process was designed to seek out the most suitable configuration of the hyperparameters listed in Table 1. Table 1 illustrates model hyperparameters, along with their types, boundaries, and their primary function. Therefore, boundaries have been chosen as follows: 10-200 neurons per layer for maintaining consistency with the small dataset that was used earlier; the number of hidden layers chosen is between 1 and 3 so that unnecessary depth can be avoided. The learning rate and α are chosen within widely used ranges by Adam between 10^{-4} and 10^{-1}.

Table 1. Model Hyperparameter

Hyperparameter	Type	Min. value	Max. value	Primary Function
Number of layers	Discrete (Integer)	1	3	Network Depth (number of hidden layers)
Layer 1 size	Discrete (Integer)	10	200	Width of the first hidden layer
Layer 2 size	Discrete (Integer)	10	200	Width of the second hidden layer
Layer 3 size	Discrete (Integer)	10	200	Width of the third hidden layer
Learning rate	Continuous (Real)	1/10000	0.1	Initial step size for the Adam optimizer
L2 regularization (α)	Continuous (Real)	1/10000	0.1	L2 Regularization parameter (Weight Decay)

6.4 Population Initialization

Nine different POAs have been benchmarked, three Foundational: DE, GA, PSO and six Sports-Inspired: GoldenBall, MVPA, SLCA, TikiTaka, TugOfWar, WorldCup. Each of the nine was run under the same unified optimization framework implemented in Python. For fairness and reproducibility, all runs underwent the same random initialization and used the same parameter settings. Table 2 lists the starting values of parameters for all optimizers.

Table 2. Population Parameters Initialization

Parameter	Description	Value
(N_{ITER})	Maximum number of iterations	20
(POP_{SIZE})	Population size	10
Random seed	Initialization for reproducibility	42
Dataset	Benchmark (Iris)	150 samples

Each optimizer initialized a population of candidate hyperparameter vectors sampled uniformly within the predefined search bounds. Subsequently, each algorithm evolved this population by iteratively minimizing the neural network's loss function through its distinct exploration-exploitation mechanism.

All algorithms have the same conditions: an identical population size of 10, an identical maximum iteration of 20, an identical random seed value of 42, identical search bounds, identical 3-fold CV fitness evaluation, and identical stagnation-based early stopping. This ensures a fair comparison, without initialization bias or unequal evaluation budgets.

6.5 Objective Function Formulation

A specific definition of the optimization objective has been adopted throughout this work for all sports-inspired algorithms. All optimizers aim to minimize the neural network classification error on the validation set, as a proxy for model generalization. Each candidate solution, composed of a tuple of hyperparameters, has then been evaluated with respect to the objective function defined by equation (1).

$$Minimize\ f(\theta) = 1 - Accuracy_{val}(\theta) \tag{1}$$

where θ is the vector of hyperparameters that involves the number of hidden layers, neurons per layer, learning rate, and L2 regularization coefficient. During each iteration, the candidate solutions are evaluated using a 3-fold cross-validation strategy to ensure robustness and avoid overfitting. This update of the population by the optimizer is intended to minimize f(θ), thereby maximizing the model's validation accuracy.

Each candidate hyperparameter vector is assessed with 3-fold CV. The averaged validation accuracy over the folds is used as the fitness value. This prevents overfitting on a single split and yields smoother optimization trajectories for the population-based algorithms. For small datasets, such as Iris, 3-fold CV is more stable than using a single train/validation split. Further reducing the usable training data by using a separate validation set would increase the variance. CV provides a more consistent estimate of fitness.

6.6 Optimization Algorithms

In applying each of the nine algorithms, an iterative update of candidate solutions is performed through unique strategies, ranging from team-based interactions to local neighborhood exploitation and global-best influence. At each iteration, FE is applied with the aim of efficiently evaluating the fitness.

6.7 Termination Criteria

This process continued with optimization until any of the following termination criteria were fulfilled:

- Maximum Iteration Criterion: The algorithm converged due to the limit on iteration being reached in this instance, defined by NITER = 20.
- Stagnation criterion (convergence check): In this regard, the algorithm was claimed to converge and, hence, terminate the execution prior to utilizing the total budget if the global best fitness did not improve in ten consecutive iterations.
- These termination conditions are chosen such that a trade-off between computational efficiency and the quality of the solution is achieved while avoiding unnecessary function evaluations after convergence.

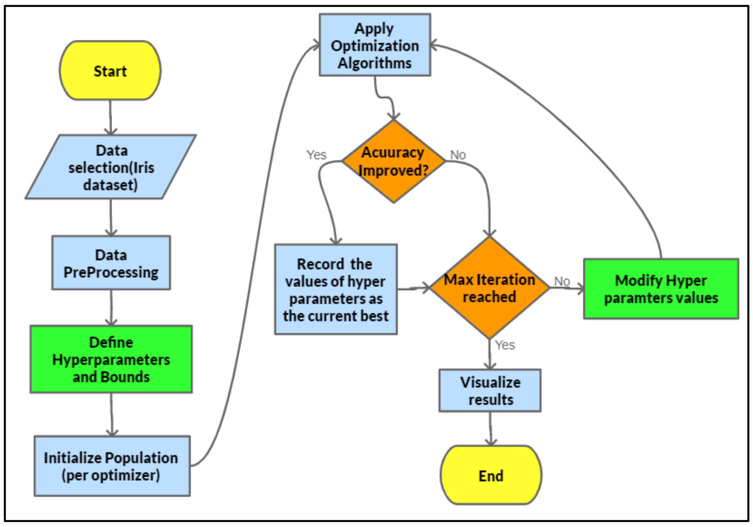
Start
Data selection(Iris dataset)
Data PreProcessing
Define Hyperparameters and Bounds
Initialize Population (per optimizer)
Apply Optimization Algorithms
Acuuracy Improved?
Yes
No
Record the values of hyper parameters as the current best
Max Iteration reached
No
Yes
Modify Hyper paramters values
Visualize results
End

Figure 1. Proposed Approach

6.8 Performance Metrics

The performance of each optimizer was quantitatively evaluated using five main metrics:

- Best Classification Accuracy (%): The highest validation accuracy achieved by the neural network across all iterations.
- Functional Evaluations (FE): The number of times the objective function was called, representing computational effort.
- Number of Iterations to Convergence: The total number of iterations required until the termination criterion was met.
- Computation Time (s): Total runtime required to complete the optimization process.
- Total neurons in optimized MLP This allows for jointly measuring the effectiveness, accuracy and efficiency, FE, iteration count, and time for each of the various sports-inspired algorithms.

6.9 Visualization and Analysis

The bar and scatter plots are used for result visualization to study the distribution of the hyperparameters, the outcome of functional evaluation, and performance comparisons across algorithms.

7. Results and Discussion

This section provides the empirical results for nine algorithms applied to the problem of optimizing neural network hyperparameters for classification. The entire HPO process executed and subsequent neural network training using the Google Colab environment. Standard Python libraries were used throughout, such as scikit-learn for preprocessing and MLP Classifier for the deep learning model, in order to ease replication of these experiments as much as possible. The objective function employs 3-fold cross-validation (cv=3) to assess the fitness of a given hyperparameter vector. The average cross-validation accuracy represents the sole fitness function f(x) that the nine optimization algorithms were tasked with.

Each optimizer searched for the best configuration of the number of hidden layers, the number of neurons per layer (n1, n2, n3), learning rate, and L2 regularization (α). All the runs followed the same termination criteria: the maximum of 20 iterations, or there is no improvement in best accuracy for ten consecutive iterations. The results are analyzed based on various dimensions: classification accuracy, hidden layer configuration, learning parameters, functional evaluations, and runtime.

7.1 Results

Table 3 reports the optimization outcomes of all algorithms, containing the following columns: best achieved accuracy, number of hidden layers ("# of layers"), number of neurons in the first hidden layer ("n1"), number of neurons in the second hidden layer ("n2") (if the value is zero, this means the target algorithm achieved the best performance using only one layer), number of neurons in the third hidden layer ("n3") (if the value is zero, this means the target algorithm achieved the best performance using only one layer), learning rate, L2 regularization ("α"), the total number of neurons in the three hidden layers ("Total neurons"), the number of iterations to reach convergence ("# iterations"), the total number of function evaluations ("#FE"), and the computation time in seconds ("Time(s)").

Table 3. Comparative Results of the Optimizers

Optimizer	Best Accuracy	# Of Layers	n1	n2	n3	learning rate	alpha	Total neurons	# Iterations	# FE	Time (s)
MVPA	0.993333	2	125	71	0	0.035305	0.092154	196	20	230	49.85
PSO	0.986667	2	88	83	0	0.005169	0.088406	171	13	140	25.45
Golden Ball	0.986667	2	156	125	0	0.020424	0.091076	281	20	210	44.41
SLCA	0.986667	3	119	164	138	0.012234	0.098344	421	18	550	146.05
DE	0.980000	1	95	0	0	0.086241	0.100000	95	13	140	20.42
TikiTaka	0.980000	2	112	108	0	0.032231	0.093151	220	11	120	25.93
TugOf War	0.973333	2	168	66	0	0.003535	0.093956	234	10	110	17.41
GA	0.973333	2	168	66	0	0.003535	0.093956	234	10	220	39.81
WorldCup	0.973333	2	168	66	0	0.003535	0.093956	234	10	110	20.14

7.2 Analysis of Results

7.2.1 Accuracy

The results of comparative accuracy shown in Figure 2 reflect that MVPA obtained the best accuracy at 99.33%, leading all the optimizers. PSO, SLCA, and Golden Ball achieved close accuracies of about 98.67%, performing stably and strongly. This confirms that team-based adaptive cooperation is superior to simple robot cooperation; hence, MVPA and SLCA are effective. Figure 2 shows the best classification accuracy for every optimizer. The figure indicates that MVPA returns the best accuracy, while the World Cup algorithm has the worst performance.

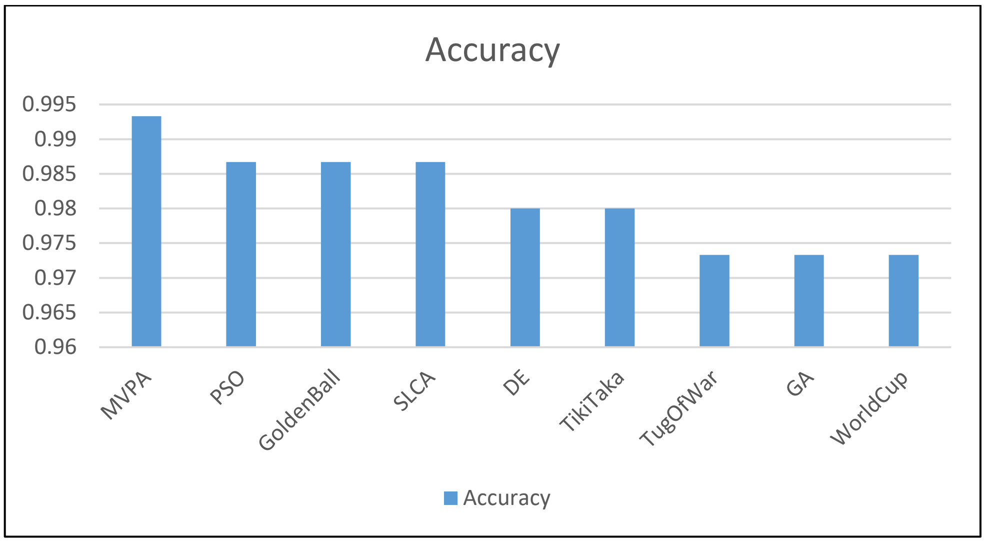

Figure 2. Best Classification Accuracy for Each Optimizer

7.2.2 Hidden Layer Structures

Figure 3 shows the optimal number of hidden layers per optimizer: SLCA needs three hidden layers, while DE requires only one hidden layer. Figure 4 illustrates the number of neurons in each layer of the model, ranging from 60 to 170 neurons per layer. Finally, Figure 5 displays the total number of neurons across all hidden layers, with the DE optimizer having a smaller total due to its single hidden layer, while SLCA has a maximum total because it utilizes three hidden layers.

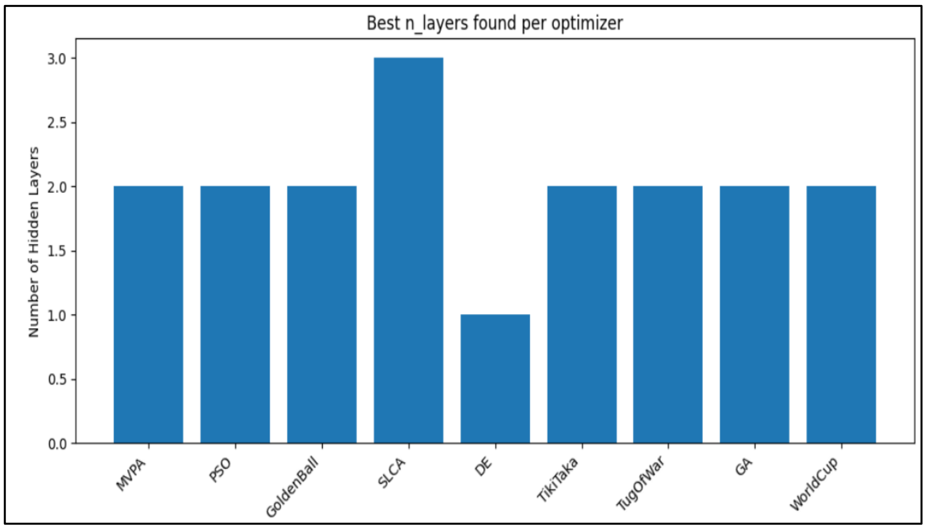

Figure 3. Best Number of Hidden Layers Per Optimizer

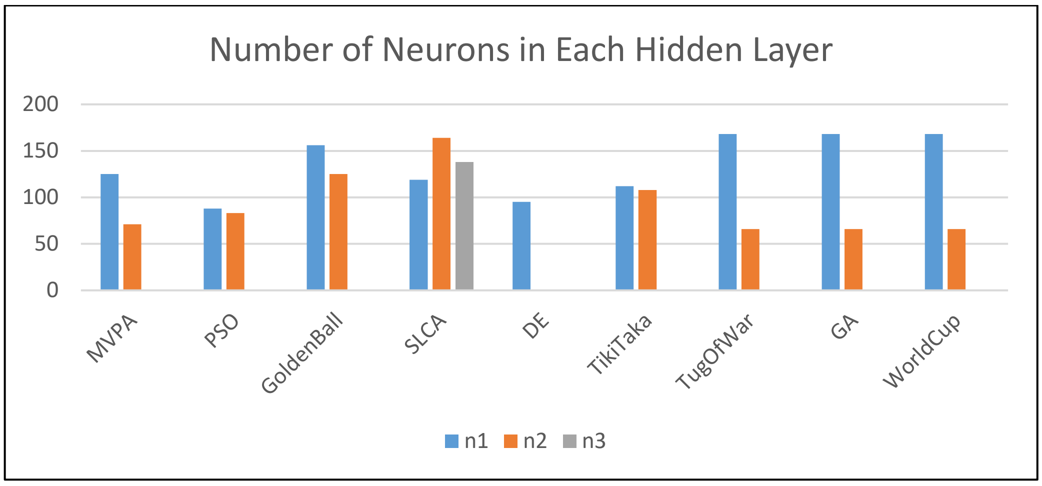

Figure 4. The Number of Neurons in Each Hidden Layer

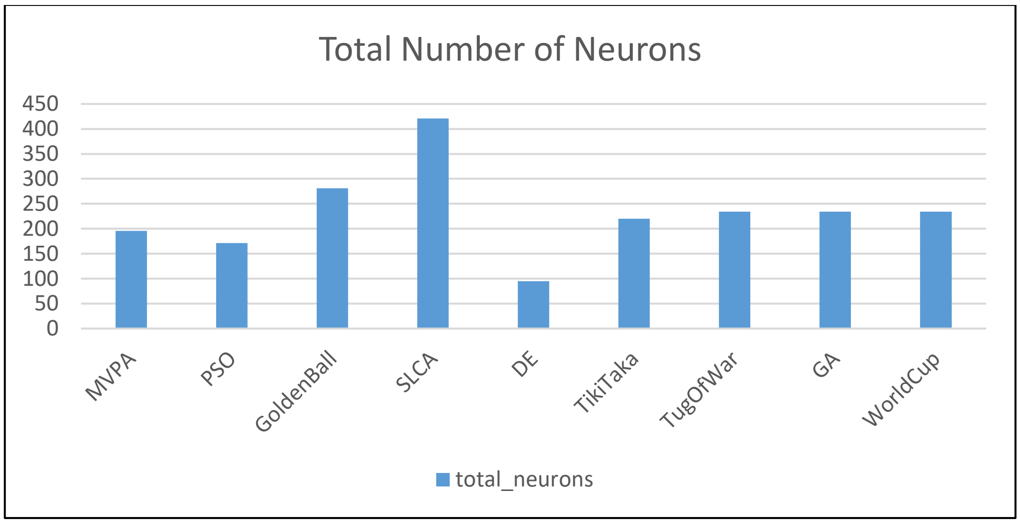

Figure 5. Total Number of Neurons Across All the Hidden Layers

This distribution of the best structural parameters found per optimizer can be summarized in the following points:

- MVPA, PSO, and GoldenBall converged on two-layer architectures, which suggests sufficient representational capacity without overfitting.
- SLCA, by contrast, had the best performance with three hidden layers and a total of 421 neurons. This indicates that deeper structures may sometimes help improve learning when guided by effective optimization.
- DE, with one layer of neurons (95), reached 98% accuracy and demonstrated the efficiency of the algorithm for smaller models as well.

7.2.3 Learning Rate and Regularization (α)

Figure 6 provides the optimal learning rate values per optimizer, while Figure 7 provides the optimal alpha value per optimizer, on a log scale.

- Most optimizers converged within the range of moderate-to-low learning rates, 0.003–0.035, which indicates that smoother gradient updates improved generalization.
- Alpha regularization strengths ranged from approximately 0.09 to 0.10, which stands to reason since model simplicity should be maintained to avoid overfitting during the metaheuristic search.

These consistent patterns of parameters across optimizers emphasize that sports-inspired search processes naturally identify balanced trade-offs between convergence speed and model regularization.

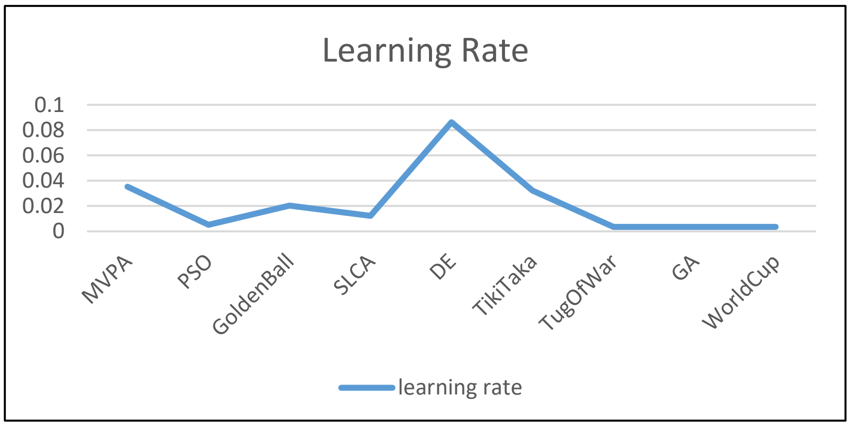

Figure 6. Learning Rate Per Optimizer

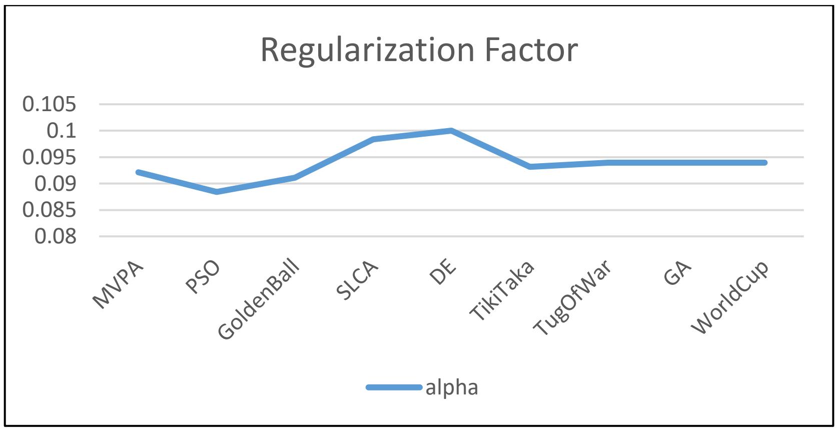

Figure 7. Regularization Factor Per Optimizer

7.2.4 Correlation Between Neuron Count and Accuracy

Figure 8 shows the best accuracy vs total neurons for the algorithms tested. There is a nonlinear but positive relation between total neuron count and classification accuracy up to around 250 neurons. After that, gains plateau, especially for SLCA and Golden Ball, suggesting that returns for model complexity beyond a certain threshold are limited.

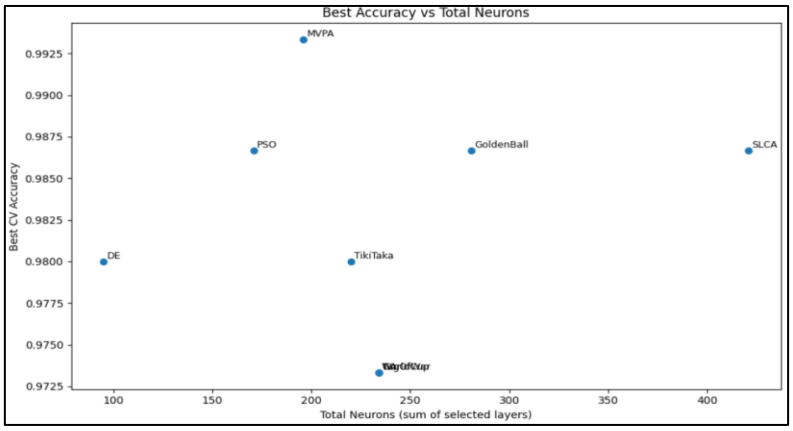

Figure 8. Best Accuracy vs Total Number of Neurons

7.2.5 Hyperparameter–Accuracy Relationships

Figure 9 shows pairwise scatter plots that explore dependencies between each hyperparameter and the achieved accuracy. MVPA's and SLCA's solutions cluster in regions of high neuron counts and moderate learning rates, reinforcing their strong exploration-exploitation balance.

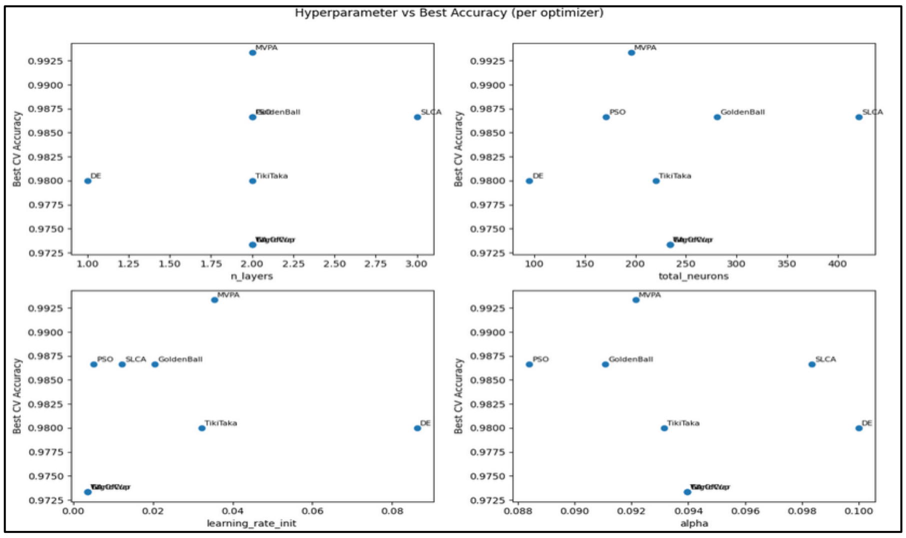

Figure 9. Relation between Hyperparameters Values vs the Accuracy Per Optimizer

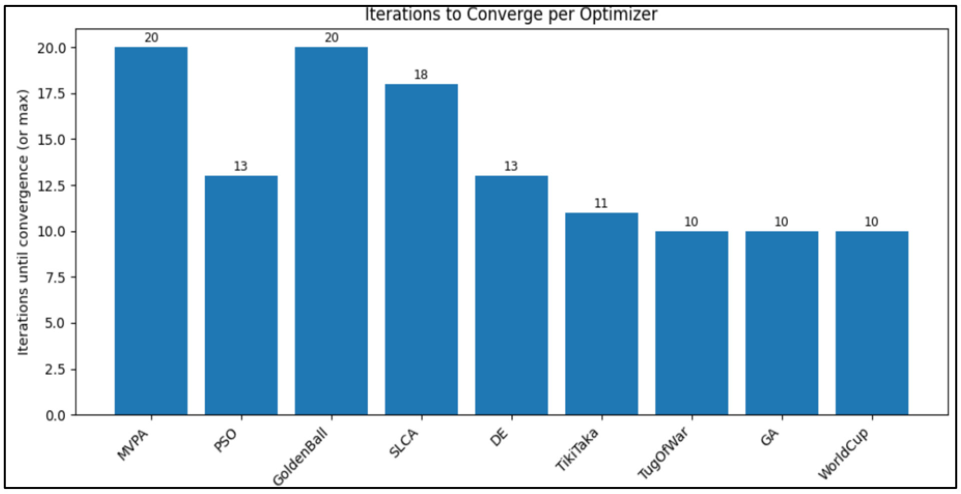

Figure 10. Iteration to Convergence Per Optimizer

7.2.6 Convergence and Functional Evaluations

Figure 10 shows the number of iterations to convergence per optimizer, and Figure 11 shows the number of functional evaluations (FEs) per optimizer. From Figures 10 and 11, the following points were noted:

- Maximum iteration limits were reached (20) for MVPA and Golden Ball, with particular emphasis on their thorough search.
- SLCA, though the slowest among the three with 18 iterations, required the highest FE count because of its multi-team evaluation mechanism: 550 FEs, or approximately 146 seconds.
- Tug of War, GA, and WCO converged within 10 iterations; this implies early stabilization but at the cost of slightly lower accuracy (~97.33%).

These differences reflect the trade-off between quality and computational cost, with deeper competition models yielding better accuracies for higher runtimes, from MVPA to SLCA.

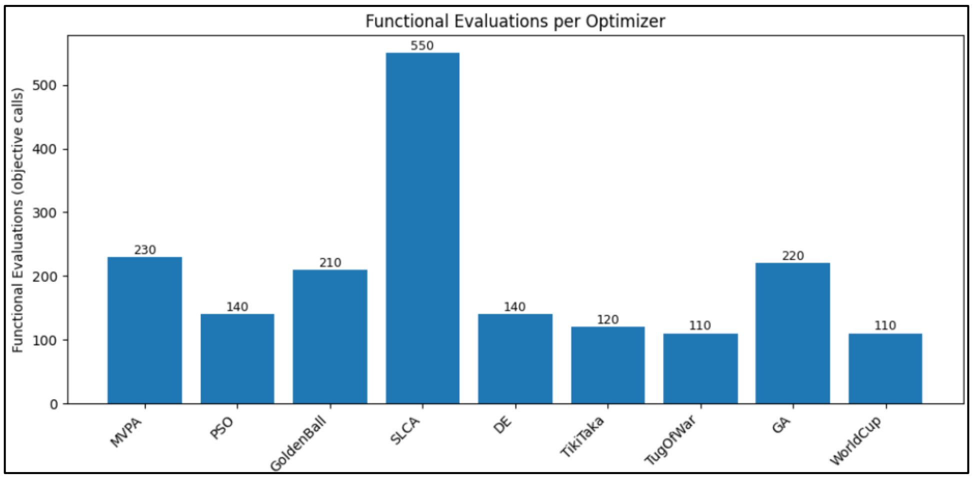

Figure 11. Functional Evalutions Per Optimizer

7.3 Discussion

Accuracy and Robustness: MVPA had the best balance of accuracy and stability, closely followed by SLCA and GoldenBall. Their success is due to player evaluation, hierarchical team organization, and adaptive role switching that enhance global search efficiency.

Efficiency: Algorithms such as TOW and DE converged in ≤13 iterations and are, therefore, more suitable for real-time or resource-limited applications.

Depth–Complexity Trade-off: SLCA's three-layer configuration provided marginal accuracy improvement but incurred heavy computational costs, emphasizing that architectural depth must be balanced against training time. SLCA uses more neurons compared to the other optimizers. This is because SLCA involves repeated reinforcement of strong teams and multi-offspring generation. The aforementioned mechanisms bias the search toward high-capacity models, hence explaining their tendency to favor larger networks.

Exploration–Exploitation Balance: Sports-inspired methods are effective in maintaining adaptive learning dynamics. MVPA's evaluation strategy and SLCA's league competition both mitigate premature convergences.

Generalization: While differing in specific architecture, all top algorithms-maintained test accuracy above 97%, reflecting robust generalization to unseen data.

7.4 Limitations

The benchmark assumes a restricted architecture search a maximum depth of three hidden layers, a maximum width of 200 neurons per layer, fixed activation, ReLU, small dataset-Iris. These choices limit generalization to large-scale models such as CNNs, transformers, or deep MLPs used in modern applications.

7.5 Overview of Results

MVPA showed the best performance overall with 99.33% accuracy, but it had a moderately high computational cost.

- SLCA yielded the same accuracy of 98.67% while having the highest number of functional evaluations, 550.
- PSO and GoldenBall presented an outstanding trade-off between performance and efficiency.
- GA, TOW, and WCO were computationally efficient but slightly less accurate.

8. Conclusion

The current research tested six sports-inspired optimization algorithms, namely MVPA, SLCA, GBA, TikiTaka Algorithm, TOW, and WCO, for the neural network hyperparameter optimization problem vis-à-vis PSO, DE, and GA. MVPA gave the best result with 99.33% classification accuracy, establishing a good trade-off between exploration and exploitation of the solution space with considerable efficiency in computation. SLCA achieved the second-best result with 98.67%. GBA converged quickly along with PSO, making it suitable for real-time applications. However, DE, TOW, and WCO yielded poor accuracies in the range of 97-98%, but their convergence is fast, making them suitable for time-bound applications. In this way, the investigation results confirm that sports-inspired metaheuristics perform well in the optimization of neural networks and often outperform conventional algorithms. Future studies will be directed towards hybridization for efficiency enhancement, scalability for complex datasets, adaptive mechanisms for convergence, and much deeper neural networks incorporating various architectures.

References

[1] Liao, Lizhi, Heng Li, Weiyi Shang, and Lei Ma. "An Empirical Study of the Impact of Hyperparameter Tuning and Model Optimization on the Performance Properties of Deep Neural Networks." ACM Transactions on Software Engineering and Methodology (TOSEM) 31, no. 3 (2022): 1-40.

[2] [Sarsa, Sami, Juho Leinonen, and Arto Hellas. "Empirical Evaluation of Deep Learning Models for Knowledge Tracing: Of Hyperparameters and Metrics on Performance and Replicability." arXiv preprint arXiv:2112.15072 (2021).

[3] Ying, Hejie, Mengmeng Song, Yaohong Tang, Shungen Xiao, and Zimin Xiao. "Enhancing Deep Neural Network Training Efficiency and Performance Through Linear Prediction." Scientific Reports 14, no. 1 (2024): 15197.

[4] Narayanan, Ramachandran, and Narayanan Ganesh. "A Comprehensive Review of Metaheuristics for Hyperparameter Optimization in Machine Learning." Metaheuristics for Machine Learning: Algorithms and Applications (2024): 37-72.

[5] Bacanin, Nebojsa, Catalin Stoean, Miodrag Zivkovic, Miomir Rakic, Roma Strulak-Wójcikiewicz, and Ruxandra Stoean. "On the Benefits of Using Metaheuristics in the Hyperparameter Tuning of Deep Learning Models for Energy Load Forecasting." Energies 16, no. 3 (2023): 1434.

[6] Tian, Zhirui, and Mei Gai. "Football Team Training Algorithm: A Novel Sport-Inspired Meta-Heuristic Optimization Algorithm for Global Optimization." Expert Systems with Applications 245 (2024): 123088.

[7] Alhijawi, Bushra, and Arafat Awajan. "Genetic algorithms: Theory, Genetic Operators, Solutions, and Applications." Evolutionary Intelligence 17, no. 3 (2024): 1245-1256.

[8] Dhar, Sandipan, Anuvab Sen, Aritra Bandyopadhyay, Nanda Dulal Jana, Arjun Ghosh, and Zahra Sarayloo. "Differential Evolution Algorithm Based Hyper-Parameters Selection of Convolutional Neural Network for Speech Command Recognition." arXiv preprint arXiv:2310.08914 (2023).

[9] Dhar, Sandipan, Anuvab Sen, Aritra Bandyopadhyay, Nanda Dulal Jana, Arjun Ghosh, and Zahra Sarayloo. "Differential Evolution Algorithm Based Hyper-Parameters

Selection of Convolutional Neural Network for Speech Command Recognition." arXiv preprint arXiv:2310.08914 (2023).

[10] Worawattawechai, Tanawat, Boonyarit Intiyot, Chawalit Jeenanunta, and William G. Ferrell Jr. "A Learning Enhanced Golden Ball Algorithm for the Vehicle Routing Problem with Backhauls and Time Windows." Computers & Industrial Engineering 168 (2022): 108044.

[11] Bouchekara, H. R. E. H. "Most Valuable Player Algorithm: A Novel Optimization Algorithm Inspired from Sport." Operational Research 20, no. 1 (2020): 139-195.

[12] Moosavian, Naser, and Babak Kasaee Roodsari. "Soccer League Competition Algorithm, A New Method for Solving Systems of Nonlinear Equations." International journal of intelligence science 4, no. 01 (2013): 7.

[13] Ab. Rashid, Mohd Fadzil Faisae. "Tiki-taka Algorithm: A Novel Metaheuristic Inspired by Football Playing Style." Engineering Computations 38, no. 1 (2021): 313-343.

[14] Kaveh, Ali. "Tug of war optimization." In Advances in Metaheuristic Algorithms for Optimal Design of Structures, Cham: Springer International Publishing, 2021, 467-503.

[15] Razmjooy, Navid, Mohsen Khalilpour, and Mehdi Ramezani. "A New Meta-Heuristic Optimization Algorithm Inspired by FIFA World Cup Competitions: Theory and Its Application in PID Designing for AVR System." Journal of Control, Automation and Electrical Systems 27, no. 4 (2016): 419-440.

[16] Dahl, George E., Frank Schneider, Zachary Nado, Naman Agarwal, Chandramouli Shama Sastry, Philipp Hennig, Sourabh Medapati et al. "Benchmarking Neural Network Training Algorithms." arXiv preprint arXiv:2306.07179 (2023).

[17] Syaharuddin, Syaharuddin, Fatmawati Fatmawati, and Herry Suprajitno. "The Formula Study in Determining the Best Number of Neurons in Neural Network Backpropagation Architecture with Three Hidden Layers." Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi) 6, no. 3 (2022): 397-402.

[18] Sharkawy, Abdel-Nasser. "The Effect of Increasing Hidden Layers on the Performance of the Deep Neural Network: Modelling, Investigation, and Evaluation." (2024).

[19] Sun, Ruo-Yu. "Optimization for Deep Learning: An Overview." Journal of the Operations Research Society of China 8, no. 2 (2020): 249-294.

[20] Iiduka, Hideaki. "Appropriate Learning Rates of Adaptive Learning Rate Optimization Algorithms for Training Deep Neural Networks." IEEE Transactions on Cybernetics 52, no. 12 (2021): 13250-13261.

[21] Yang, Mei, Ming K. Lim, Yingchi Qu, Xingzhi Li, and Du Ni. "Deep Neural Networks with L1 and L2 Regularization for High Dimensional Corporate Credit Risk Prediction." Expert Systems with Applications 213 (2023): 118873.

[22] Raiaan, Mohaimenul Azam Khan, Sadman Sakib, Nur Mohammad Fahad, Abdullah Al Mamun, Md Anisur Rahman, Swakkhar Shatabda, and Md Saddam Hossain Mukta. "A Systematic Review of Hyperparameter Optimization Techniques in Convolutional Neural Networks." Decision Analytics Journal 11 (2024): 100470.

[23] El-Hassani, Fatima Zahrae, Meryem Amri, Nour-Eddine Joudar, and Khalid Haddouch. "A New Optimization Model for MLP hyperparameter Tuning: Modeling and Resolution by Real-Coded Genetic Algorithm." Neural Processing Letters 56, no. 2 (2024): 105.

[24] Salem, Hend S., Mohamed A. Mead, and Ghada S. El-Taweel. "Particle Swarm Optimization-Based Hyperparameters Tuning of Machine Learning Models for Big COVID-19 Data Analysis." Journal of Computer and Communications 12, no. 3 (2024): 160-183.

[25] Wong, Tzu-Tsung, and Po-Yang Yeh. "Reliable Accuracy Estimates From K-Fold Cross Validation." IEEE Transactions on Knowledge and Data Engineering 32, no. 8 (2019): 1586-1594.