Elephant Herding For SVM Classifier Optimisation

Fatiha BARRADE, Fatima NOUTFI

Keywords. *optimization algorithms, swarm intelligence, Elephant Herding Optimization (EHO), Support Vector Machines (SVM), parameter optimization,*

Introduction

In the realm of problem-solving, optimization algorithms serve as indispensable tools for tackling complex challenges across diverse domains. These algorithms play a pivotal role in finding the best possible solution from a vast array of potential options, often in situations where traditional methods fall short. By systematically exploring and evaluating different configurations, optimization algorithms offer a means to optimize various parameters, objectives, or constraints, enabling informed decision-making and resource allocation.

Swarm Intelligence (SI) stands out as a particularly intriguing and effective approach within the realm of optimization algorithms. Inspired by the collective behavior observed in natural systems, SI algorithms mimic the cooperative strategies employed by groups of animals to navigate complex environments and solve challenging problems. This approach draws parallels to the concept of emergent intelligence, wherein individual agents interact locally to produce global behavior patterns that lead to efficient solutions.

At the heart of Swarm Intelligence lies the idea of decentralized control and self-organization, where individual agents, often referred to as particles, agents, or individuals, follow simple rules and interact with one another and their environment to collectively optimize a predefined objective function. Through iterative processes of exploration and exploitation, swarm-based algorithms are capable of efficiently searching solution spaces, often outperforming traditional optimization techniques in terms of solution quality and convergence speed.

One notable exemplar of Swarm Intelligence is the Elephant Herding Optimization (EHO) algorithm, which draws inspiration from the behavior of elephant herds in nature. Elephants exhibit complex social dynamics, communication patterns, and decision-making processes within their herds, which enable them to navigate their environments effectively. EHO algorithm encapsulates these characteristics by modeling individual elephants as agents within a population and simulating their interactions to guide the search for optimal solutions.

EHO operates on the principles of collaboration, communication, and adaptation, mirroring the behavior of elephants as they coordinate movements, share information, and adapt to changing environmental conditions. By harnessing these principles, EHO offers a robust and efficient framework for solving optimization problems across various domains. Its ability to emulate the collective intelligence of elephant herds makes it well-suited for addressing complex, dynamic, and multi-dimensional optimization challenges.

One particularly promising application of EHO lies in optimizing the parameters of Support Vector Machines (SVM), a powerful machine learning algorithm used for classification and regression tasks. By integrating EHO with SVM parameter optimization, researchers and practitioners can enhance the performance and generalization capabilities of SVM models, leading to improved accuracy and efficiency in real-world applications. Moreover, the adaptability and scalability of EHO make it well-suited for addressing a wide range of optimization problems beyond SVM parameter tuning, spanning fields such as engineering, finance, healthcare, and beyond

1 Collective Intelligence

Collective Intelligence (CI) is a burgeoning field at the intersection of artificial intelligence and social dynamics, aiming to harness the collective wisdom and problem-solving capabilities of groups. At its core, CI leverages the synergy of diverse individuals to achieve outcomes that surpass the capabilities of any single participant.

1.1 Definition

CI can be defined as the capacity of a group to solve a wide range of problems more effectively than individuals or isolated machines. It encompasses the aggregation, coordination, and processing of distributed knowledge, often resulting in emergent behaviors and solutions that are not readily apparent from the contributions of individual mem-

bers.

1.2 Why We Need Collective Intelligence

In today's interconnected and rapidly evolving world, many problems are too multifaceted and dynamic to be effectively tackled by individuals or traditional problem-solving methods alone. These challenges often span multiple domains, involve diverse stakeholders with differing perspectives and expertise, and require timely responses to evolving circumstances.

CI harnesses the collective capabilities of groups to navigate such complexities. By bringing together individuals with diverse backgrounds, knowledge, and skills, CI ensures that a wide range of perspectives and insights are considered when addressing complex problems. This diversity not only enriches the pool of available solutions but also enhances the quality of decision-making by mitigating biases and blind spots that may exist in individual viewpoints.

Moreover, CI facilitates collaborative problem-solving processes that leverage the collective intelligence of the group. Through open dialogue, information sharing, and collaborative sense-making, participants can collectively generate innovative solutions that may not have been conceivable by any single individual working in isolation. This collaborative approach fosters creativity, fosters a culture of continuous learning, and promotes a sense of ownership and commitment among participants.

Furthermore, CI enables real-time adaptation to changing circumstances and emergent challenges. In today's fast-paced and interconnected world, problems can evolve rapidly, and new complexities can arise unexpectedly. CI frameworks provide the agility and flexibility needed to respond effectively to these dynamic conditions. By leveraging distributed decision-making processes and adaptive feedback loops, CI enables groups to quickly iterate and adjust their strategies in response to new information or changing conditions.

Overall, CI offers a powerful framework for tackling the complex challenges of our interconnected world. By pooling together diverse perspectives, skills, and insights, CI enables more robust decision-making, problem-solving, and innovation, ultimately empowering groups to address multifaceted problems with agility and effectiveness.

1.3 Principles of Collective Intelligence

The principles underlying CI are rooted in the study of natural systems, such as swarms, flocks, and social communities, where collective behaviors emerge from simple interactions among autonomous agents. These principles include:

- **Distributed Autonomy**: Empowering individuals or agents with autonomy while facilitating coordinated interactions and information exchange.
- **Localized Communication**: Emphasizing communication channels and protocols that enable efficient exchange of information among neighboring entities.
- **Self-Organization**: Encouraging adaptive and decentralized organization, where system-level behaviors emerge from the interactions of autonomous agents without central control.
- **Robustness and Adaptivity**: Designing systems that exhibit resilience to disruptions and can adapt to changing conditions or inputs.
- **Scalability**: Ensuring that CI systems can scale seamlessly with the size and complexity of the problem domain, leveraging parallelism and distributed processing.

By embracing these principles, CI frameworks strive to emulate the collective intelligence observed in nature, offering promising avenues for addressing real-world challenges across various domains.

2 Support Vector Machine (SVM) Classifier

2.1 Introduction

While logistic regression is effective for binary classification tasks, Support Vector Machine (SVM) classifiers offer a powerful alternative for both binary and multiclass classification problems. SVMs are particularly well-suited for scenarios where the data is not linearly separable, as they can implicitly map the input features into a higher-dimensional space where linear separation becomes feasible. In this section, we delve into the principles of SVM classifiers, their training process, and their versatility in handling complex classification tasks.

2.2 Theoretical Basis of SVMs

The primary goal in pattern classification is to develop a model that maximizes performance on the training data. Traditional training methods aim to construct models that correctly classify each input-output pair within its respective class. However, overly fitting the classifier to the training data can lead to memorization rather than generalization, thus compromising the classifier's ability to perform well on unseen data.

The main motivation behind Support Vector Machines (SVMs) is to separate multiple classes in the training set with a surface that maximizes the margin between them. In essence, SVMs aim to maximize the generalization ability of a model. This objective is aligned with the Structural Risk Minimization (SRM) principle, which seeks to minimize a bound on the generalization error of a model. This stands in contrast to the philosophy of empirical risk minimization, which typically focuses on minimizing the mean squared error on the training data.

2.3 Training Process of SVM Classifier

The training process of the Support Vector Machine (SVM) classifier involves finding the optimal hyperplane parameters that maximize the margin between different classes while minimizing the classification error.

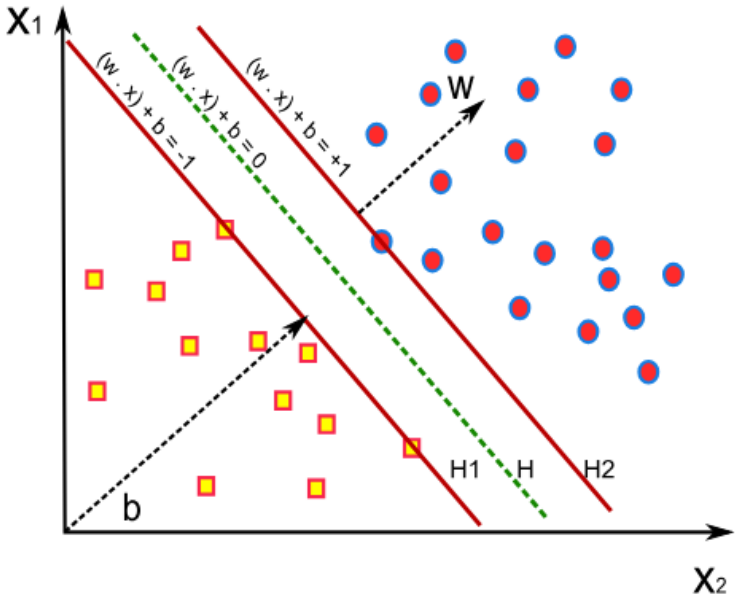

Figure 1: Best Classifier

2.3.1 Objective Function

The objective of SVM training is to minimize the classification error while maximizing the margin. This is typically formulated as an optimization problem with the following objective function:

$$\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n} \max(0, 1 - y_i(\mathbf{w} \cdot \mathbf{x}_i + b))$$

where:

- $\mathbf{w}$ is the weight vector perpendicular to the hyperplane,
- b is the bias term,
- C is the regularization parameter that controls the trade-off between maximizing the margin and minimizing the classification error,
- n is the number of training examples,
- y_i is the true class label,
- $\mathbf{x}_i$ is the feature vector of the i-th training example.

The first term in the objective function represents the regularization term to penalize large weights, while the second term is the hinge loss function that measures the classification error.

2.3.2 Optimization

The optimization problem defined by the objective function can be solved using various optimization techniques, including:

- **Gradient Descent**: Iteratively update the weights and bias in the direction of the negative gradient of the objective function.
- **Stochastic Gradient Descent (SGD)**: Update the weights and bias using a randomly selected subset of training examples at each iteration.
- **Quadratic Programming**: Formulate the optimization problem as a quadratic programming (QP) problem and use specialized solvers to find the optimal solution.
- **Sequential Minimal Optimization (SMO)**: A specialized algorithm for solving the dual formulation of the SVM optimization problem, particularly efficient for large datasets.

The optimization process aims to find the optimal values of $\mathbf{w}$ and b that minimize the objective function while satisfying the constraints imposed by the margin and the classification error.

2.3.3 Kernel Trick

The kernel trick in SVM allows handling non-linearly separable data by implicitly transforming the feature space using kernel functions. The mathematical formulation of the kernel trick can be described as follows:

Given two input feature vectors x and x', the kernel function K computes the inner product of the transformed feature vectors in a higher-dimensional space without explicitly computing the transformation. Mathematically, this can be expressed as:

$$K(x, x') = \phi(x) \cdot \phi(x')$$

Where:

- $K(x, x')$ is the kernel function that computes the similarity between two input feature vectors x and x'.
- $\phi(x)$ and $\phi(x')$ are the transformations of the input feature vectors x and x', respectively, into a higher-dimensional space.

The choice of kernel function determines the mapping of the input features into the higher-dimensional space. Common kernel functions include:

1. **Polynomial Kernel**:
$$K(x, x') = (x \cdot x' + c)^d$$
where c is a constant and d is the degree of the polynomial.

2. **Radial Basis Function (RBF) Kernel**:
$$K(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right)$$
where σ is a parameter that controls the width of the Gaussian.

3. **Sigmoid Kernel**:
$$K(x, x') = \tanh(\alpha x \cdot x' + c)$$

where α and c are constants.

These kernel functions enable SVM to effectively handle non-linearly separable data by implicitly transforming the feature space into a higher-dimensional space where linear separation is possible.

In summary, the SVM classifier offers a robust framework for classification tasks, leveraging the principles of margin maximization and regularization to achieve accurate predictions even in challenging scenarios. The training process involves optimizing the objective function using specialized techniques, while the kernel trick enables SVMs to handle non-linearly separable data effectively.

2.4 Relevance to SVM Optimization:

The integration of Elephant Herding Optimization (EHO) into the optimization process of Support Vector Machines (SVMs) holds significant promise for enhancing SVM performance. EHO offers a robust and efficient approach to explore the complex parameter space of SVMs, facilitating the identification of optimal hyperparameters and model configurations. By harnessing the collective intelligence and adaptability inspired by elephant herds, EHO can effectively navigate high-dimensional search spaces and overcome local optima, thereby improving the generalization capability and classification accuracy of SVMs. Moreover, EHO's ability to parallelize search processes and handle non-convex landscapes makes it particularly well-suited for optimizing SVMs, which often involve nonlinear decision boundaries and complex optimization landscapes. Thus, the incorporation of EHO into SVM optimization strategies has the potential to unlock new levels of performance and scalability in machine learning tasks.

3 Elephant Herding Optimization

3.1 Introduction

The Elephant Herding Optimization (EHO) algorithm represents a cutting-edge approach to optimization inspired by the intricate social structures and behaviors of elephant herds. Originating from the pioneering work of Gai-Ge Wang et al. in 2015, EHO harnesses the collective intelligence observed in elephant herds to tackle complex optimization problems across various domains.

Elephants, renowned for their sophisticated familial bonds and hierarchical organiza-

tion, serve as the primary inspiration behind EHO. Within elephant herds, matriarchal figures lead tight-knit family groups, fostering profound clan bonding among members. The remarkable emotional depth exhibited by elephants, including mourning rituals for deceased loved ones, underscores the complexity of their social dynamics.

In developing the EHO algorithm, researchers drew parallels between the leadership structure and decision-making processes observed in elephant herds and the optimization process. In essence, EHO simulates the movement and interaction of individual elephants within clans, guided by matriarchal leaders and influenced by male elephants representing suboptimal solutions.

3.2 Algorithm Description

The Elephant Herding Optimization (EHO) algorithm mirrors the social dynamics of elephant herds, where individuals collaborate and adapt to navigate their environment effectively. By integrating these principles into optimization processes, EHO offers a robust and efficient framework for solving complex optimization problems.

The complex herding behavior of elephants is mathematically modeled within the EHO algorithm. The basic EHO framework defines a set of rules to search for both global and local solutions, as outlined below:

1. **Fixed Clan Size:** The number of elephants in each clan remains constant. If an elephant leaves the clan, a new elephant or calf can replace its position.
2. **Male Elephant Separation:** In each generation/iteration, a fixed number of male elephants leave their family groups to live in isolation.
3. **Matriarchal Leadership:** The elephants within each clan operate under the leadership of their respective matriarch.

To update the location of elephants in each generation, EHO employs two position updating operators: clan updating and separation operators. These operators divide the EHO method into two distinct phases, each playing a crucial role in the optimization process.

3.2.1 Position Update of Elephants in a Clan

In this phase, the positions of elephants remaining within the clan are updated. Each elephant's position within a clan is influenced by the matriarch's leadership. Specifically, the position of the jth elephant in clan c is updated using the following equation:

$$p_{t+1,jc} = p_{t,jc} + \alpha \times (p_{\text{best}} - p_{t,jc}) \times r$$

Here, $p_{t+1,jc}$ and $p_{t,jc}$ represent the updated and previous positions in generations $t+1$ and t respectively. α is a scaling factor between $[0, 1]$, determining the influence of the matriarch. p_{best} is the position of the leader elephant holding the best fitness in the clan, and $r \in [0, 1]$ is a uniform distribution.

The position of matriarch elephants is updated differently, using the equation:

$$p_{t+1,jc} = \beta \times p_{\text{center},c}$$

where $\beta \in [0, 1]$ is a scaling factor influenced by the clan's central position, $p_{\text{center},c}$. The center of a clan c is determined by averaging the positions of all elephants in the clan.

3.2.2 Separation of Male Elephants from the Clan

Male elephants leave their family groups to live solitary lives or with male groups as they grow up. The separation process is mathematically modeled in the EHO algorithm, where elephants with the worst fitness in each clan depart from their respective clans. The new position of the jth worst elephant in clan c is determined as:

$$p_{t+1,\text{worst},jc} = p_{\min,c} + \text{rand} \times (p_{\max,c} - p_{\min,c} + 1)$$

Here, $p_{\min,c}$ and $p_{\max,c}$ represent the lower and upper bounds of elephant individuals in clan c, and rand $\in [0, 1]$ represents a uniform stochastic distribution.

3.3 Pseudo-Code for EHO

The following pseudo-code presents the Elephant Herding Optimization (EHO) algorithm for solving optimization problems inspired by the herding behavior of elephants.

Algorithm 1 EHO Algorithm

Input parameters: Objective Function $OF(\cdot)$, Number of clans N, Number of elephants in each clan n_c, Scaling factors α, β, Maximum number of generations $G_{\max}$, Lower and upper bounds for each variable/clan $[p_{\min,c}, p_{\max,c}]$
Output: Optimal solution
Initialization: Randomly generate initial positions for all elephants in each clan
for each j-th elephant **do**
    for each c-th clan **do**
        Generate random position: $p_{j,c} = p_{\min,c} + (p_{\max,c} - p_{\min,c}) \times \text{rand}$
        Evaluate fitness: $\text{Fitness}_j = OF(p_{j,c})$
    end for
end for
Set generation counter: $t = 1$
while $t \leq G_{\max}$ **do**
    Perform clan updating operator
    Perform separating operator
    Evaluate fitness of the population with newly updated positions
    Increment generation counter: $t = t + 1$
end while

This pseudo-code outlines the steps to implement the EHO algorithm, encompassing the initialization, position updating, and termination conditions for the optimization process.

4 Application Of EHO in SVM

The Elephant Herding Optimization (EHO) algorithm can be adapted for classification optimization, particularly in the context of Support Vector Machine (SVM) classifiers. Here, elephant locations are interpreted as SVM parameters within a selected feature set, while elephant fitness corresponds to the average classification accuracy across multiple cross-validation folds.

4.1 Algorithmic Framework

Algorithm 2 outlines the framework of the EHO-SVM classifier. The algorithm begins with the initialization of parameters and the generation of initial population locations, representing SVM parameters or selected features. It then iterates through generations,

applying the clan updating and separating operators, evaluating population fitness, and identifying the best elephant with the highest fitness (classification accuracy).

Algorithm 2 EHO-SVM Approach

Input: Training sets (Folds) $(T_1, T_2, \ldots, T_n)$
Input: Validation sets (Folds) $(V_1, V_2, \ldots, V_n)$
Output: Classification accuracy
Initialization:
Generation counter $t \leftarrow 1$
Initialize population locations (SVM, Kernel parameters / Selected Features)
Evaluate population fitness (g)
while $g <$ MaxGen **do**
    Sort all elephants according to fitness
    Apply clan updating operator
    Apply elephant separating operator
    Evaluate population fitness (g)
    Find best elephant with highest fitness (Classification accuracy)
    $g \leftarrow g + 1$
end while

4.2 Fitness Function

The fitness function plays a crucial role in optimization algorithms by quantifying the quality of each solution. In this work, classification accuracy serves as the fitness criterion. The accuracy value is computed as the average accuracy over all folds in the cross-validation strategy, as shown in Equation 6.

$$f(i,j) = \frac{1}{n} \sum_{k=1}^{n} Acc_{i,j,k} \tag{1}$$

The folllwoing algorithm illustrates the fitness calculation process using the SVM classifier. It involves training the SVM on each training set using the parameters and features corresponding to the elephant's location and evaluating the validation accuracy.

Algorithm 3 Evaluate Elephant Population Fitness

Input: Training sets (Folds) $(T_1, T_2, \ldots, T_n)$
Input: Validation sets (Folds) $(V_1, V_2, \ldots, V_n)$
Input: Population number (j)
Output: Total accuracy
Initialize: $Acc \leftarrow 0$
for each elephant i in population j **do**
    Get elephant location $Loc_{i,j}$
    for each training set $T_i \in T_1, T_2, T_3$ **do**
        SVM parameters $P \leftarrow \text{GetParameters}(Loc_{i,j})$
        Selected Features $F \leftarrow \text{GetFeatures}(Loc_{i,j})$
        Train SVM on T_i using P, F
        Validation Accuracy $V \leftarrow$ Validate V_i
        $Acc \leftarrow Acc + V$
    end for
end for
Total Accuracy $\leftarrow \frac{Acc}{n}$
Fitness $\leftarrow$ Total Accuracy

This section highlights the integration of EHO with SVM classifiers, leveraging EHO's optimization capabilities to enhance classification accuracy through parameter and feature optimization.

5 Adaptive Mutation Enhanced Elephant Herding Optimization (AMEHO)

The Adaptive Mutation Enhanced Elephant Herding Optimization (AMEHO) algorithm is a variant of the Elephant Herding Optimization (EHO) algorithm, designed to address the limitations of traditional optimization algorithms. AMEHO introduces adaptive mutation mechanisms to overcome issues such as premature convergence and exploration-exploitation trade-offs, making it effective for a wide range of optimization tasks.

5.1 Why AMEHO is Effective

- **Adaptive Mutation:** AMEHO incorporates a mutation mechanism that adapts during the algorithm's run based on the performance and failure rates of the

features being optimized. This adaptive mutation strategy enhances exploration and exploitation capabilities, allowing the algorithm to effectively navigate the search space and find optimal solutions.

- **Clan-based Optimization:** AMEHO organizes the search process into clans, mimicking the social structure of elephant herds. Each clan consists of individual elephants representing potential solutions. By organizing individuals into clans, AMEHO facilitates information sharing and collaboration among solutions, leading to more efficient exploration of the search space.
- **Dynamic Updating Operators:** AMEHO employs dynamic updating operators for clan management. These operators ensure a balanced exploration-exploitation trade-off by continuously adjusting the positions of individual elephants within each clan. By dynamically updating the positions of individuals based on the current state of the search, AMEHO effectively balances exploration of new regions with exploitation of promising solutions.
- **Fitness-guided Selection:** AMEHO uses fitness-guided selection mechanisms to guide the search process towards regions of the search space with higher potential for optimization. By prioritizing individuals with higher fitness values, AMEHO directs the search towards promising solutions, leading to faster convergence and improved optimization performance.

5.2 Algorithm Overview

AMEHO operates within a population of individual elephants, each representing a potential solution to an optimization problem. The algorithm iteratively evolves these solutions through a series of clan updating and separating operations, guided by the fitness of each solution.

5.2.1 Clan Updating Operator

In the clan updating operator, each individual elephant adjusts its position based on the position of the clan matriarch and the clan center. The update equation is as follows:

$$x_{i,j,t+1} = x_{i,j,t} + \alpha(m_{i,t} - x_{i,j,t+1}) + \beta(c_{i,t} - x_{i,j,t+1}) + \gamma * r$$

Where:

- $x_{i,j,t}$ is the position of individual elephant j in clan i at time t.

- $m_{i,t}$ is the position of the clan matriarch in clan i at time t.
- $c_{i,t}$ is the clan center position of clan i at time t.
- α, β, and γ are control parameters.
- r is a random vector.

5.2.2 Separating Operator

The separating operator replaces the worst individual elephant in each clan with a new individual generated based on the clan matriarch's position. The update equation is as follows:

$$x_{i,worst,t+1} = x_{min} + (x_{max} - x_{min}) * rand * x_{adapt}$$

Where:

- x_{min} and x_{max} are the lower and upper bounds of the feature space.
- $rand$ is a random number generated uniformly.
- x_{adapt} is the adaptive mutation parameter calculated based on the fitness function.

5.2.3 Calculation of Adaptive Mutation Parameter

The adaptive mutation parameter x_{adapt} is determined based on the fitness function. It reflects the performance and failure rates of the features being optimized. Specifically, it adjusts the mutation rate during the algorithm's run to enhance exploration and exploitation capabilities.

The calculation of x_{adapt} can be represented as follows:

$$x_{adapt} = \left(1 + \frac{f_{max} - f_{avg}}{f} \cdot \exp(\gamma \cdot N(0,1))\right)^{-1}$$

Where:

- f signifies the fitness value of the feature.
- f_{max} signifies the best fitness value of the current generation.
- f_{avg} represents the average fitness value of the current generation.

- γ is the learning rate that defines the likelihood of elephants to wander (walk in a random direction) and controls the variation speed.
- $N(0, 1)$ is a random vector drawn from a normal distribution.

This calculation adjusts the mutation rate based on the difference between the best and average fitness values, as well as the learning rate γ, ensuring that the mutation rate adapts to the current state of the search process.

5.3 Pseudocode

The pseudocode for the AMEHO algorithm is as follows:

Algorithm 4 Adaptive Mutation Enhanced Elephant Herding Optimization (AMEHO)

procedure AMEHO($MaxGen, n$) ▷ Initialize
  $t \leftarrow 1$
  while $t \leq MaxGen$ **do**
    for $i \leftarrow 1$ to $nClan$ **do** ▷ Clan operations
      Sort elephants in clan i by fitness
      $x^t_{best} \leftarrow$ position of the first elephant in clan i
      $x^t_{worst} \leftarrow$ position of the last elephant in clan i
      $m^t_i \leftarrow x^t_{best}$
      Replace the worst elephant in clan i [using the separating operator]
      for $j \leftarrow 1$ to nCi **do** ▷ Individual elephant updates
        Update position of individual elephant j in clan i [using the clan updating operator]
      end for
    end for
    $t \leftarrow t + 1$
  end while
end procedure

Conclusion

n conclusion, the integration of Elephant Herding Optimization (EHO) and its adaptive variant, Adaptive Mutation Enhanced Elephant Herding Optimization (AMEHO), into the optimization process of Support Vector Machines (SVMs) presents a promising approach for enhancing SVM performance. By leveraging the collective intelligence and adaptability inspired by elephant herds, these algorithms offer robust and efficient frameworks for exploring the complex parameter space of SVMs.

EHO and AMEHO provide effective solutions for overcoming challenges such as premature convergence, exploration-exploitation trade-offs, and local optima, thereby improving the generalization capability and classification accuracy of SVMs. Their ability to parallelize search processes and handle non-convex landscapes makes them particularly well-suited for optimizing SVMs, which often involve nonlinear decision boundaries and complex optimization landscapes.

Through the emulation of elephant herds' social dynamics and hierarchical organization, EHO and AMEHO offer innovative optimization strategies that can unlock new levels of performance and scalability in machine learning tasks. By incorporating these algorithms into SVM optimization strategies, researchers and practitioners can potentially achieve superior results in classification tasks across various domains.

References

[1] A comprehensive survey on support vector machine classification: Applications, challenges and trends. https://www.sciencedirect.com/science/article/pii/S0925231220307153

[2] Jurafsky, D., & Martin, J. H. (2023). Speech and Language Processing. Copyright © 2023. All rights reserved. Draft of February 3, 2024. Retrieved from:

[3] Elephant Herding Optimization: Variants, Hybrids, and Applications https://us.sagepub.com/en-us/nam/applied-logistic-regression-analysis/book220724

[4] Support Vector Machine Optimized by Elephant Herding Algorithm for Erythemato-Squamous Diseases Detectionhttps://www.sciencedirect.com/science/article/pii/S1877050917327035

[5] Combining Support Vector Machine and Elephant Herding Optimization for Cardiac Arrhythmias https://www.researchgate.net/publication/325922364_Combining_Support_Vector_Machine_and_Elephant_Herding_Optimization_for_Cardiac_Arrhythmias

[6] Slowik, A. (2021). Swarm Intelligence Algorithms.