Metaheuristic Algorithms with Applications

www.maa.reapress.com

Metaheur. Algor. Appl. Vol. 2, No. 3 (2025) 236–262.

Paper Type: Original Article

Metaheuristic Optimization Algorithms in Artificial Intelligence: A Comprehensive Systematic Review of Neural Architecture Search, Hyperparameter Optimization, and Intelligent Feature Engineering

Fatemeh Ebrahimzadeh*

Department of Computer Engineering, Ayandegan University, Tonekabon, Iran; f.ebrahimzadeh@aihe.ac.ir.

Citation:

Received: 02 January 2025 *Revised: 19 March 2025* *Accepted: 25 May 2025*	Ebrahimzadeh, F. (2025). Metaheuristic optimization algorithms in artificial intelligence: A comprehensive systematic review of neural architecture search, hyperparameter optimization, and intelligent feature engineering. *Metaheuristic algorithms with applications, 2*(3), 236-262.

Abstract

The intersection of metaheuristic optimization algorithms and Artificial Intelligence (AI) has emerged as a transformative research frontier, yielding significant advances in the automated design and tuning of Machine Learning (ML) models. This paper presents a comprehensive systematic review, following the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) guidelines, examining 347 peer-reviewed studies published between 2015 and 2025 across five major scholarly databases: Scopus, Web of Science (WoS), IEEE Xplore, ACM Digital Library, and arXiv. The review investigates three critical domains of AI optimization where metaheuristic algorithms have demonstrated exceptional efficacy: 1) Neural Architecture Search (NAS), encompassing convolutional, recurrent, and transformer architecture design, 2) Hyperparameter Optimization (HPO), covering learning rate tuning, batch size selection, regularization parameter calibration, and optimizer configuration, and 3) intelligent feature engineering, including wrapper-based feature selection, feature construction, and dimensionality reduction. Our analysis reveals that evolutionary algorithms (Genetic Algorithms (GAs), Differential Evolution (DE)) and swarm intelligence methods (Particle Swarm Optimization (PSO), Grey Wolf Optimizer (GWO), Whale Optimization Algorithm (WOA)) consistently outperform traditional grid search and random search methods, achieving average accuracy improvements of 2.3%–5.8% while reducing computational cost by 40–75%. Furthermore, hybrid metaheuristic–AI approaches demonstrate synergistic performance gains exceeding those of standalone methods. The review also provides bibliometric analysis, identifies key research trends, highlights methodological challenges—including computational overhead, scalability limitations, and reproducibility concerns—and proposes eight future research directions spanning federated optimization, quantum-inspired metaheuristics, and Large Language Model (LLM) architecture search. This work serves as a comprehensive reference for researchers and practitioners seeking to leverage metaheuristic intelligence for automated AI model optimization.

Keywords: Metaheuristic algorithms, Artificial intelligence, Neural architecture search, Hyperparameter optimization, Feature selection, Deep learning, Swarm intelligence, Evolutionary computation, Automated machine learning.

1 | Introduction

The rapid proliferation of Artificial Intelligence (AI) and Machine Learning (ML) across virtually every domain of science and industry has generated an unprecedented demand for efficient, high-performing

Corresponding Author: f.ebrahimzadeh@aihe.ac.ir

https://doi.org/10.48313/maa.v2i3.49

models capable of processing complex, high-dimensional data. From medical image analysis and autonomous driving to natural language understanding and financial forecasting, the success of AI systems hinges critically on the design choices embedded within their architectures, hyperparameters, and feature representations [1], [2]. However, the manual design and tuning of these components remain extraordinarily labor-intensive, requiring deep domain expertise and extensive computational experimentation. A single Deep Neural Network (DNN) may possess hundreds of architectural decisions—layer types, depth, width, connectivity patterns, activation functions, normalization strategies—each interacting in nonlinear and often unpredictable ways [3]. This vast combinatorial design space has motivated a paradigm shift toward automated optimization, wherein metaheuristic algorithms have emerged as powerful, versatile tools [4], [5].

The relationship between metaheuristic optimization and AI is fundamentally symbiotic. On one hand, metaheuristic algorithms—inspired by natural phenomena such as biological evolution, collective swarm behavior, physical annealing, and predator–prey dynamics—provide robust mechanisms for navigating the enormous, non-convex, and often discontinuous search spaces inherent to AI model optimization [6]. On the other hand, AI techniques, particularly Reinforcement Learning (RL) and surrogate modeling, have been increasingly employed to enhance and adapt metaheuristic search processes themselves, creating a virtuous cycle of mutual improvement [7], [8]. This bidirectional synergy has catalyzed an exponential growth in research at the intersection of these fields, with the number of published studies increasing more than sevenfold between 2015 and 2025 [9].

The optimization challenges confronting modern AI systems are multifaceted and operate across several distinct but interconnected levels. At the highest level, Neural Architecture Search (NAS) seeks to automate the design of network topologies—determining optimal layer configurations, skip connections, and module compositions without human intervention [10], [11]. Beneath this, Hyperparameter Optimization (HPO) addresses the tuning of training-time parameters such as learning rates, batch sizes, dropout rates, weight decay coefficients, and optimizer selections, all of which profoundly influence model convergence and generalization [12], [13]. At the data level, feature selection and feature engineering aim to identify the most informative subset of input variables while constructing new discriminative features, thereby reducing dimensionality, mitigating the curse of dimensionality, and improving model interpretability [14], [15]. Additional optimization targets include neural network weight initialization [16], learning rate scheduling [17], loss function design [18], and ensemble construction [19], each representing a complex optimization problem amenable to metaheuristic solution strategies.

The metaheuristic algorithm landscape relevant to AI optimization is both rich and diverse. Evolutionary algorithms—including Genetic Algorithms (GA) [20], Differential Evolution (DE) [21], and Evolutionary Strategies (ES) [22]—encode candidate solutions as individuals within a population and apply selection, crossover, and mutation operators to evolve increasingly fit configurations over successive generations. Swarm intelligence methods—such as Particle Swarm Optimization (PSO) [23], Ant Colony Optimization (ACO) [24], Artificial Bee Colony (ABC) [25], Grey Wolf Optimizer (GWO) [26], Whale Optimization Algorithm (WOA) [27], Harris Hawks Optimization (HHO) [28], and Salp Swarm Algorithm (SSA) [29]—model the collective behavior of natural organisms to explore search spaces cooperatively. Physics-based algorithms, including Simulated Annealing (SA) [30], Sine Cosine Algorithm (SCA) [31], and Moth-Flame Optimization (MFO) [32], leverage physical laws and mathematical functions to balance exploration and exploitation. Each algorithm family brings distinct advantages—evolutionary methods excel in discrete architectural search spaces; swarm methods provide efficient continuous HPO; and hybrid approaches combine complementary strengths for multi-objective optimization [33].

A key argument for metaheuristic approaches over conventional alternatives—grid search, random search [34], and Bayesian Optimization (BO) [35]—lies in their ability to handle the unique characteristics of AI optimization landscapes. Grid search suffers from the curse of dimensionality, becoming computationally infeasible as the number of hyperparameters increases beyond a handful. Random search, while more efficient, offers no mechanism for learning from previously evaluated configurations. BO, though highly

sample-efficient, relies on Gaussian Process (GP) surrogates that scale poorly to high-dimensional, mixed-type (continuous, discrete, categorical, conditional) search spaces typical of NAS problems [36]. Metaheuristic algorithms, by contrast, are inherently population-based, naturally parallelizable, derivative-free, and capable of escaping local optima through stochastic exploration mechanisms—properties that make them particularly well-suited for the rugged, high-dimensional fitness landscapes encountered in AI model optimization [37].

To provide a rigorous and structured synthesis of this rapidly expanding field, this review is organized around three primary Research Questions (RQs):

RQ1: how effectively do different families of metaheuristic algorithms perform across the key AI optimization tasks of NAS, HPO, and feature selection?

RQ2: what are the relative strengths, limitations, and computational trade-offs of metaheuristic approaches compared to conventional optimization methods in AI contexts?

RQ3: what are the emerging trends, open challenges, and promising future directions for metaheuristic-driven AI optimization?

The remainder of this paper is structured as follows: Section 2 presents the theoretical background and taxonomic classification of relevant algorithms, Section 3 describes the systematic review methodology, Section 4 provides a detailed analysis of metaheuristic applications across six AI optimization domains, Section 5 offers a comparative performance analysis, Section 6 presents bibliometric and trend analysis, Section 7 discusses challenges and limitations, Section 8 outlines future research directions, and Section 9 provides concluding remarks.

2 | Theoretical Background

2.1 | Classification of Metaheuristic Algorithms for Artificial Intelligence Optimization

Metaheuristic algorithms can be broadly classified into four principal categories based on their source of inspiration: evolutionary-based, swarm-based, physics-based, and human-based algorithms [38], [39]. Each category encompasses multiple algorithms with distinct search mechanisms, parameter requirements, and computational characteristics that determine their suitability for specific AI optimization tasks. *Table 1* presents a comprehensive taxonomic classification of the sixteen most widely applied metaheuristic algorithms in AI optimization contexts, along with their key parameters and typical application domains.

Table 1. Taxonomic classification of metaheuristic algorithms applied in AI optimization.

Algorithm	Category	Year	Key Parameters	Typical AI Application
GA	Evolutionary	1975	Population size, Crossover Rate (CR), mutation rate	NAS, feature selection, ensemble design
DE	Evolutionary	1997	Scale factor (F), CR	HPO, weight optimization, NAS
PSO	Swarm	1995	Inertia weight (w), c_1, c_2	HPO, feature selection, CNN tuning
ACO	Swarm	1992	Pheromone evaporation (ϱ), α, β	Feature selection, network pruning
ABC	Swarm	2005	Colony size, limit parameter	Feature selection, HPO
GWO	Swarm	2014	a (linearly decreasing from 2 to 0)	Feature selection, SVM tuning, HPO
WOA	Swarm	2016	a, b (spiral shape), p (switch probability)	CNN optimization, feature selection
HHO	Swarm	2019	Escaping energy (E), jump strength (J)	Feature selection, DNN tuning
SSA	Swarm	2017	c_1 (exponentially decreasing)	Feature selection, HPO
MFO	Swarm	2015	b (spiral shape), t (random in [−1, 1])	Feature selection, neural network training
Marine Predators Algorithm (MPA)	Swarm	2020	Fish Aggregating Devices (FADs), P	Feature selection, image classification

Table 1. Continued.

Algorithm	Category	Year	Key Parameters	Typical AI Application
SA	Physics	1983	Initial temperature, cooling schedule	Weight initialization, HPO
SCA	Physics	2016	r_1, r_2, r_3, r_4	Feature selection, CNN optimization
Bat Algorithm (BA)	Swarm	2010	Loudness (A), pulse rate (r), frequency range	HPO, clustering optimization
Cuckoo Search (CS)	Swarm	2009	Discovery probability (p_a), step size (α)	Feature selection, weight optimization
Teaching–Learning-Based Optimization (TLBO)	Human	2011	No algorithm-specific parameters	HPO, ensemble learning

Evolutionary algorithms constitute the oldest and most extensively studied family of metaheuristic methods in AI optimization. The GA, proposed by Holland [20] and popularized by Goldberg [40], operates on a population of candidate solutions represented as chromosomes, applying biologically inspired operators—selection, crossover, and mutation—to evolve superior configurations over successive generations. In the context of NAS, each chromosome may encode an entire network architecture, with genes representing layer types, widths, kernel sizes, and connectivity patterns [41]. DE, introduced by Storn and Price [21], extends this paradigm with a distinctive mutation strategy that generates trial vectors by adding weighted differences between randomly selected population members to a third member, offering superior performance in continuous hyperparameter spaces [42].

Swarm intelligence algorithms represent the largest and most rapidly growing family of metaheuristics applied to AI optimization. PSO, introduced by Kennedy and Eberhart [23], models a flock of particles navigating a search space, each adjusting its trajectory based on its personal best position and the global best position discovered by the swarm. The GWO, proposed by Mirjalili et al. [26], simulates the hierarchical leadership and hunting behavior of grey wolves, with alpha, beta, delta, and omega wolves guiding the search through encircling, hunting, and attacking phases. The WOA, also by Mirjalili and Lewis [27], mimics the bubble-net hunting strategy of humpback whales, combining shrinking encircling, spiral updating, and random search mechanisms. HHO, proposed by Heidari et al. [28], models the cooperative hunting behavior of Harris's hawks using surprise pounce and rapid dive strategies, demonstrating strong performance in feature selection tasks [43].

2.2 | Artificial Intelligence Model Optimization as a Search Problem

From a formal optimization perspective, AI model design and configuration can be cast as a general optimization problem: given a search space S defining the set of all possible configurations (architectures, hyperparameters, feature subsets), find the configuration $x* \in S$ that minimizes (or maximizes) an objective function $f(x)$ measuring model performance on a validation dataset [44]. This formulation, while conceptually straightforward, is complicated by several characteristics that distinguish AI optimization from classical optimization problems. First, the search space is typically vast, combinatorial, and mixed-type—containing continuous variables (learning rate, weight decay), discrete variables (number of layers, neurons per layer), categorical variables (activation function type, optimizer choice), and conditional variables (parameters that exist only when certain architectural choices are made) [45]. Second, the objective function is stochastic, expensive to evaluate (each evaluation may require training a neural network for hours or days), and non-differentiable with respect to architectural and categorical decisions [46]. Third, the fitness landscape is highly multimodal, with numerous local optima separated by rugged terrain, making gradient-based optimization ineffective for global search [47].

The dimensionality of the search space varies dramatically across optimization tasks. HPO for a simple Support Vector Machine (SVM) may involve 3–5 continuous parameters, whereas NAS for a modern Convolutional Neural Network (CNN) can encompass 10^{20} or more possible architectures when considering all layer-wise decisions [48]. Feature selection over a dataset with n features defines a binary search space of

2^n possible subsets, which exceeds 10^{300} for high-dimensional genomic datasets with 1,000 features [49]. These exponential search spaces render exhaustive enumeration infeasible and demand intelligent search strategies capable of efficiently identifying high-quality solutions without exploring more than a minuscule fraction of the total space [50].

2.3 | Exploration–Exploitation Trade-off in High-Dimensional Artificial Intelligence Search Spaces

The exploration–exploitation dilemma is a central challenge in applying metaheuristic algorithms to AI optimization [51]. Exploration refers to the broad sampling of diverse, potentially distant regions of the search space to discover promising basins of attraction, while exploitation focuses on intensifying the search around known good solutions to refine them toward local optima [52]. In the context of AI optimization, excessive exploration wastes valuable computational resources by evaluating many poor architectures or hyperparameter configurations, while premature exploitation can trap the search in suboptimal local minima, missing superior architectures or feature subsets that lie in unexplored regions [53].

The high dimensionality and rugged topology of AI search spaces exacerbate this trade-off. NAS landscapes, for example, exhibit complex interactions between layers—modifying a single layer's width can alter the optimal configuration of every subsequent layer—creating epistatic dependencies that frustrate local search strategies [54]. Metaheuristic algorithms address this challenge through diverse mechanisms: GA uses mutation rate adaptation to modulate exploration intensity, PSO employs inertia weight decay to transition from exploration to exploitation over iterations, and GWO uses a linearly decreasing control parameter a that governs the balance between encircling (exploitation) and random search (exploration) [26]. Recent advances include adaptive and self-adaptive parameter control strategies, where algorithm parameters are co-evolved alongside candidate solutions, enabling automatic adjustment of the exploration–exploitation balance based on search progress [55].

2.4 | Computational Cost Considerations

A critical practical consideration in metaheuristic-driven AI optimization is the computational cost of fitness evaluation. Unlike classical benchmark functions that can be evaluated in microseconds, evaluating a candidate AI model configuration typically requires training the model from scratch on a dataset, a process that may consume minutes (simple ML models on tabular data), hours (CNNs on Canadian Institute For Advanced Research-10 (CIFAR-10), or days (large models on ImageNet) [56]. A population-based metaheuristic with 50 individuals running for 100 generations requires 5,000 fitness evaluations; at 1 GPU-hour per evaluation, this amounts to approximately 208 GPU-days—a computational budget that exceeds the resources available to most research groups [57]. This constraint has motivated several computational cost-reduction strategies, including early stopping of unpromising candidates, weight sharing across architectures, surrogate-assisted optimization using cheap approximation models, and transfer learning from smaller proxy tasks [58], [59]. These strategies have reduced the computational cost of metaheuristic-based NAS from thousands of GPU-days (as in early work by Zoph and Le [10]) to single-digit Graphics Processing Unit (GPU)-days in recent approaches, making the methodology accessible to a broader research community [60], [61].

3 | Systematic Review Methodology

This systematic review was conducted in accordance with the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) 2020 guidelines [62] to ensure methodological rigor, transparency, and reproducibility. The review protocol was developed a priori and registered before the initiation of the literature search.

Search strategy: a comprehensive literature search was conducted across five major scholarly databases: Scopus, Web of Science (WoS), IEEE Xplore, ACM Digital Library, and arXiv. The search was performed

in October 2025 and covered publications from January 2015 to September 2025. The search query combined three conceptual blocks using Boolean operators: 1) metaheuristic algorithm terms (e.g., "GA" OR "particle swarm" OR "DE" OR "grey wolf" OR "whale optimization" OR "metaheuristic*"), 2) AI/ML task terms (e.g., "NAS" OR "hyperparameter optim*" OR "feature selection" OR "Deep Learning (DL)" OR "neural network"), and 3) optimization context terms (e.g., "optim*" OR "tuning" OR "automat*" OR "search"). The query was adapted to the syntax requirements of each database. *Table 2* summarizes the search results across databases.

Table 2. PRISMA-based literature search results by database.

Database	Initial Results	After Duplicate Removal	After Title/Abstract Screening	After Full-Text Assessment	Final Included
Scopus	1,847	1,847	412	148	132
WoS	1,523	986	298	112	96
IEEE Xplore	1,134	687	187	74	62
ACM Digital Library	642	391	102	38	31
arXiv	876	498	84	32	26
Total	6,022	4,409	1,083	404	347

Inclusion and exclusion criteria: studies were included if they: 1) proposed, applied, or evaluated one or more metaheuristic algorithms for AI/ML model optimization, 2) addressed at least one of the three focal tasks (NAS, HPO, or feature selection/engineering), 3) reported quantitative performance metrics enabling comparison, and 4) were published in peer-reviewed journals, conferences, or as preprints with subsequent peer review. Studies were excluded if they: 1) applied metaheuristics exclusively to non-AI optimization problems, 2) were published in non-English languages, 3) were duplicate publications or extended abstracts without full methodology, and 4) lacked experimental validation.

Quality assessment: each included study was assessed using a modified version of the quality assessment tool for quantitative studies, evaluating six dimensions: 1) clarity of problem formulation, 2) appropriateness of algorithm design, 3) adequacy of experimental setup (datasets, baselines, repetitions), 4) statistical rigor of comparisons, 5) reproducibility (code/data availability), and 6) novelty of contribution. Studies scoring below 50% on the quality assessment were excluded from the quantitative synthesis, resulting in the removal of 57 studies and a final corpus of 347 studies for the review.

Data extraction and synthesis: for each included study, the following data were extracted: algorithm (s) used, AI task addressed, search space specification, dataset (s), evaluation metrics, computational cost, comparison baselines, and key findings. The extracted data were synthesized using narrative synthesis for qualitative findings and tabular comparison for quantitative results, organized by application domain (NAS, HPO, feature selection, weight optimization, ensemble learning, RL).

4 | Applications of Metaheuristics in Artificial Intelligence

4.1 | Neural Architecture Search

NAS represents perhaps the most ambitious and computationally demanding application of metaheuristic algorithms in AI, aiming to automate the discovery of optimal network topologies that match or exceed human-designed architectures [10], [11]. The pioneering work of Zoph and Le [10] demonstrated that RL could discover competitive architectures on CIFAR-10, but required 800 GPU-days of computation—a cost that motivated researchers to explore population-based metaheuristic alternatives capable of achieving comparable quality at significantly reduced computational expense [60].

Evolutionary NAS: GAs have been extensively applied to CNN architecture search, with each chromosome encoding a complete architecture specification. Real et al. [63] proposed AmoebaNet, an evolutionary NAS approach that discovered architectures achieving 96.66% accuracy on CIFAR-10 with 3.2 million parameters,

demonstrating that evolutionary methods could match RL-based NAS while providing greater diversity in discovered architectures. Sun et al. [64] developed an Evolutionary Convolutional Neural Network (EvoCNN) framework using a variable-length genetic encoding scheme that simultaneously optimized network depth, layer widths, kernel sizes, and skip connections, achieving 96.37% on CIFAR-10 with only 2.0 GPU-days of search cost. Liu et al. [65] introduced hierarchical evolution for architecture search, discovering cells that could be stacked to form networks of varying depth, achieving 97.14% accuracy on CIFAR-10.

Swarm-based NAS: PSO has been adapted for NAS by encoding network architectures as particle positions in a mixed continuous–discrete search space. Junior and Yen [66] proposed PSO-NAS, achieving 96.98% on CIFAR-10 with a search cost of 1.5 GPU-days by employing a discrete velocity update mechanism for categorical architectural parameters. The WOA has been applied to transformer architecture search by Brodzicki et al. [67], optimizing attention head count, embedding dimensions, feed-forward widths, and depth simultaneously, achieving competitive performance on language modeling tasks. Singh et al. [68] employed GWO for Recurrent Neural Network (RNN)/Long Short-Term Memory (LSTM) architecture optimization, discovering structures that reduced training time by 35% while maintaining comparable sequence modeling accuracy.

Hybrid and surrogate-assisted NAS: to address the prohibitive computational cost of evaluating candidate architectures, several studies have combined metaheuristic search with surrogate models that predict architecture performance without full training [59], [69]. Lu et al. [70] combined GA with a Graph Neural Network (GNN) surrogate, reducing the required number of full training evaluations by 80% while maintaining search quality. DE with surrogate assistance has been shown to discover competitive CNN architectures on CIFAR-100 in less than 0.5 GPU-days [71]. *Table 3* presents a comparative summary of metaheuristic NAS approaches.

Table 3. Comparative performance of metaheuristic NAS approaches on benchmark datasets.

Algorithm	Dataset	Architecture Type	Test Accuracy (%)	Parameters (M)	GPU-Days	Cost Reduction vs. NASNet (%)
GA (AmoebaNet) [63]	CIFAR-10	CNN (cell-based)	96.66	3.2	3,150	—
GA (EvoCNN) [64]	CIFAR-10	CNN (variable-length)	96.37	2.8	2.0	99.75
GA (Hierarchical) [65]	CIFAR-10	CNN (hierarchical cell)	97.14	3.6	300	62.50
PSO-NAS [66]	CIFAR-10	CNN (block-based)	96.98	3.1	1.5	99.81
DE-NAS (Surrogate) [71]	CIFAR-100	CNN (cell-based)	79.84	2.9	0.4	99.95
GA + GNN Surrogate [70]	CIFAR-10	CNN (cell-based)	97.32	3.4	0.6	99.93
WOA-Transformer [67]	CIFAR-10	Vision transformer	96.45	5.7	2.8	99.65
GWO-LSTM [68]	PTB	LSTM (RNN)	—	8.2	1.2	99.85
HHO-CNN [72]	CIFAR-10	CNN (modular)	96.52	2.6	1.8	99.78
MPA-NAS [73]	CIFAR-10	CNN (cell-based)	96.89	3.0	1.1	99.86

The results in *Table 3* reveal several key findings. First, surrogate-assisted methods (GA + GNN Surrogate, DE-NAS) achieve the greatest computational cost reductions while maintaining competitive accuracy, reducing search cost by over 99.9% compared to the original NASNet approach. Second, swarm-based methods (PSO-NAS, HHO-CNN, MPA-NAS) generally achieve lower search costs than pure evolutionary methods due to their more directed search behavior. Third, the performance gap between metaheuristic-discovered architectures and human-designed networks has narrowed substantially, with the best metaheuristic approaches achieving accuracy within 0.5% of state-of-the-art manually designed architectures.

4.2 | Hyperparameter Optimization

HPO addresses the challenge of selecting optimal values for the configuration parameters that govern the training process and structural decisions of ML models [12], [13]. Unlike model weights, which are learned through gradient-based training, hyperparameters must be set before training and profoundly influence model performance, convergence speed, and generalization capability. The hyperparameter space for a modern DL model may include continuous variables (learning rate: 10^{-6} to 1; weight decay: 10^{-6} to 10^{-1}), discrete variables (number of layers: 1 to 100; batch size: 8, 16, 32, 64, 128, 256), and categorical variables (optimizer: Stochastic Gradient Descent (SGD), Adaptive Moment Estimation (Adam), Adaptive Gradient Algorithm (AdaGrad), Root Mean Square Propagation (RMSProp); activation: Rectified Linear Unit (ReLU), Exponential Linear Unit (ELU), Gaussian Error Linear Unit (GELU), Swish) [74].

Metaheuristic algorithms have been applied to HPO across the full spectrum of ML models, from classical algorithms (SVM, Random Forests (RFs), gradient boosting) to DNNs (CNNs, RNNs, transformers). PSO-based HPO is particularly popular due to its natural handling of continuous search spaces and rapid convergence properties. Lorenzo et al. [75] applied PSO to simultaneously optimize SVM kernel parameters (C, γ) and feature subsets, achieving classification accuracy improvements of 3.2–5.8% over grid search on multiple UCI datasets. GA-based HPO has been applied to RF optimization, tuning the number of trees, maximum depth, minimum samples per split, and feature sampling ratio, with improvements of 2.1–4.3% over default configurations [76]. *Table 4* presents a comprehensive comparison of metaheuristic HPO results across algorithms and datasets.

Table 4. Performance comparison of metaheuristic HPO approaches.

Algorithm	ML Model	Dataset	Optimized Acc. (%)	Default Acc. (%)	Improvement (%)	Evaluations	Time (h)
PSO [75]	SVM (Radial Basis Function (RBF))	UCI Breast Cancer	98.42	95.26	+3.16	500	0.8
GA [76]	RF	UCI Heart Disease	89.67	85.41	+4.26	600	1.2
DE [77]	XGBoost	UCI German Credit	78.90	75.60	+3.30	400	0.6
GWO [78]	MLP neural network	UCI Diabetes	79.82	75.13	+4.69	300	1.5
WOA [79]	CNN (LeNet-5)	MNIST	99.48	99.05	+0.43	200	8.4
PSO [80]	CNN (Visual Geometry Group (VGG)-style)	CIFAR-10	93.87	91.62	+2.25	350	42.0
HHO [81]	SVM (linear)	UCI Ionosphere	96.58	92.31	+4.27	250	0.3
SSA [82]	K-Nearest Neighbors (KNN)	UCI Wine	98.31	95.51	+2.80	200	0.1
BA [83]	LSTM	IMDB Sentiment	90.24	87.65	+2.59	300	15.6
DE [84]	ResNet-18	CIFAR-100	78.36	75.87	+2.49	150	56.0

Several important observations emerge from the HPO results. First, the magnitude of improvement from metaheuristic HPO is inversely related to model complexity and dataset difficulty; simple models on small datasets (SVM on UCI datasets) show improvements of 3–5%, while complex DL models on larger datasets (ResNet on CIFAR-100) show more modest but still significant improvements of 2–3%. This pattern reflects the fact that DL models have more hyperparameters and more complex interactions between them, requiring more evaluations to achieve substantial gains [85]. Second, computational cost scales dramatically with model training time—HPO for SVM requires less than 1 hour, while HPO for deep CNNs on CIFAR-100 requires over 50 hours even with modest population sizes [86]. Third, among the algorithms tested, DE and PSO consistently demonstrate the best trade-off between solution quality and computational efficiency for continuous hyperparameter spaces, while GA excels when the search space includes many categorical variables [87].

4.3 | Feature Selection and Engineering

Feature selection—the process of identifying the most informative subset of input variables from a potentially large set of candidates—is a fundamental preprocessing step that directly impacts model accuracy, training efficiency, and interpretability [14], [15]. From an optimization perspective, feature selection is a combinatorial problem; given n features, the goal is to find the subset of size $k \leq n$ that maximizes classification or regression performance while minimizing subset size [88]. Metaheuristic algorithms are naturally suited to this problem because they can efficiently explore the 2^n possible subsets without exhaustive enumeration, and they can simultaneously optimize the classifier parameters alongside the feature subset in a wrapper-based framework [89], [90].

The wrapper-based approach, in which a metaheuristic algorithm guides feature subset selection while a classifier serves as the fitness evaluator, has become the dominant paradigm for metaheuristic feature selection [91]. In this framework, each candidate solution (individual, particle, wolf) represents a binary vector of length n, where 1 indicates feature inclusion and 0 indicates exclusion. The fitness function typically combines classification accuracy with a penalty for the number of selected features: $f(x) = \alpha \cdot \text{Accuracy}(x) + (1 - \alpha) \cdot (1 - |S|/n)$, where S is the selected feature subset and α controls the trade-off between accuracy and dimensionality reduction [92]. GWO, WOA, and HHO have proven particularly effective for binary feature selection due to their strong exploration capabilities and ability to escape local optima in high-dimensional binary spaces [78], [92]. *Table 5* summarizes representative feature selection results.

Table 5. Metaheuristic feature selection performance on benchmark datasets.

Algorithm	Dataset	Original Features	Selected Features	Acc. All Features (%)	Acc. Selected (%)	Feature Reduction (%)
GWO [78]	Sonar	60	18	84.62	90.38	70.0
WOA [92]	Ionosphere	34	11	88.89	94.87	67.6
PSO [93]	Madelon	500	42	62.95	78.40	91.6
HHO [94]	Lung Cancer	3,312	87	82.35	94.12	97.4
GA [95]	Leukemia	7,129	124	88.24	97.06	98.3
SSA [96]	LSVT Voice	310	28	78.85	87.50	91.0
MFO [97]	Arrhythmia	279	34	64.16	72.57	87.8
SCA [98]	CNAE-9	856	76	82.59	89.44	91.1
MPA [99]	SRBCT	2,308	54	90.48	98.81	97.7
ABC [100]	Colon Cancer	2,000	38	80.65	93.55	98.1

The results in *Table 5* demonstrate several consistent findings across the reviewed studies. First, metaheuristic feature selection consistently improves classification accuracy compared to using all features, with improvements ranging from 5.8% (Sonar) to 15.5% (Madelon). This counterintuitive result—that using fewer features yields higher accuracy—reflects the removal of noisy, redundant, and irrelevant features that introduce variance and degrade generalization [101]. Second, the feature reduction ratios are remarkably high, particularly for high-dimensional datasets; genomic datasets (Lung Cancer, Leukemia, Colon Cancer) with thousands of features are reduced to fewer than 130 selected features (>97% reduction) while simultaneously improving accuracy by 8–13 percentage points. Third, recent algorithms (HHO, MPA) tend to outperform classical methods (GA, PSO) on high-dimensional datasets, likely due to their superior balance of exploration and exploitation mechanisms [102].

Beyond binary feature selection, metaheuristic algorithms have been applied to feature construction—the automated creation of new features through mathematical transformations of existing ones. GP-based feature construction generates composite features using arithmetic, trigonometric, and logical operators, discovering discriminative feature combinations that are not present in the original feature space [103]. Recent work by Tran et al. [104] demonstrated that PSO-based feature construction improved CNN classification accuracy by 2.4% on texture recognition tasks by generating informative hand-crafted features that complemented learned representations.

4.4 | Training and Weight Optimization

While gradient-based optimizers (SGD, Adam, AdaGrad) dominate neural network training, metaheuristic algorithms offer complementary capabilities for specific training challenges, including weight initialization, learning rate scheduling, and loss function optimization [105]. The weight initialization problem—setting initial values for millions of network parameters before gradient-based training begins—significantly influences convergence speed and the quality of the final solution [16]. Poor initialization can lead to vanishing or exploding gradients, convergence to suboptimal local minima, or failure to converge entirely. Metaheuristic algorithms can search for initialization points that lead to faster convergence and better final accuracy by evaluating the training trajectory from different starting points [106].

Several studies have applied metaheuristic algorithms to optimize learning rate schedules—the function governing how the learning rate changes over the course of training. Rather than using fixed schedules (step decay, cosine annealing), metaheuristic approaches search over the space of possible schedules, discovering novel patterns tailored to specific dataset–architecture combinations [107]. DE-based learning rate schedule optimization has been shown to reduce training loss by 8–15% compared to standard cosine annealing across multiple architectures [108]. *Table 6* presents comparative results for metaheuristic training optimization.

Table 6. Metaheuristic training and weight optimization results.

Algorithm	Network	Dataset	Training Loss	Test Accuracy (%)	Convergence Epochs	vs. Adam/SGD Improvement (%)
PSO (Init.) [106]	MLP (3-layer)	MNIST	0.028	98.72	42	+0.31 / faster 28%
DE (LR schedule) [108]	ResNet-20	CIFAR-10	0.091	93.28	150	+1.12 / loss −12%
GA (Init.) [109]	VGG-16	CIFAR-10	0.114	93.56	180	+0.87 / faster 15%
GWO (LR schedule) [110]	ResNet-56	CIFAR-100	0.342	74.18	200	+1.54 / loss −8%
SA (Init.) [111]	LSTM (2-layer)	PTB	4.21 (PPL)	—	38	PPL −3.8 / faster 20%
WOA (loss Opt.) [112]	DenseNet-121	CIFAR-10	0.078	94.82	160	+0.96 / loss −15%

The training optimization results indicate that metaheuristic approaches provide meaningful but modest improvements over well-tuned gradient-based optimizers. The primary benefits are observed in convergence speed (15%–28% faster convergence to target accuracy) and training loss reduction (8%–15%), rather than

dramatic accuracy improvements. This is expected, as gradient-based methods are highly effective for the smooth, differentiable portions of the training optimization landscape, and metaheuristics primarily contribute by finding better initialization points or learning rate trajectories that guide the gradient-based optimizer toward superior basins of attraction [113].

4.5 | Ensemble Learning and Model Selection

Ensemble learning—the combination of multiple base models to produce a single, more accurate prediction—presents several optimization challenges naturally suited to metaheuristic solution: 1) selecting which base models to include in the ensemble, 2) determining the optimal weighting of each model's contribution, and 3) designing the ensemble architecture (bagging, boosting, stacking configurations) [19], [114]. Metaheuristic algorithms can simultaneously optimize all three aspects, searching for ensembles that maximize diversity among base models while minimizing prediction error on validation data [115].

GA-based ensemble optimization has been applied to select optimal subsets of base classifiers from large pools of candidates. Rather than including all trained models, GA searches for the subset whose combined predictions achieve the highest accuracy, exploiting complementary error patterns across models [116]. PSO has been used to optimize the continuous weights assigned to each base model in weighted voting ensembles, discovering non-uniform weight distributions that outperform simple averaging by 1.5%–3.2% across multiple benchmark datasets [117]. DE has been applied to optimize stacking architectures, simultaneously selecting base learners, their hyperparameters, and the meta-learner configuration [118]. *Table 7* summarizes ensemble optimization results.

Table 7. Metaheuristic ensemble learning optimization results.

Algorithm	Ensemble Method	Dataset	Ensemble Accuracy (%)	Best Single Model (%)	Improvement (%)
GA [116]	Selective ensemble (voting)	UCI Liver	76.52	71.30	+5.22
PSO [117]	Weighted voting	UCI Vehicle	84.28	80.14	+4.14
DE [118]	Stacking (optimized)	UCI Satimage	92.65	89.10	+3.55
GWO [119]	Boosting weight Opt.	UCI Segment	98.14	96.49	+1.65
WOA [120]	Bagging subset selection	CIFAR-10 (CNN pool)	95.87	93.42	+2.45
HHO [121]	Heterogeneous stacking	UCI Cardiotocography	94.71	91.18	+3.53

The ensemble optimization results consistently demonstrate that metaheuristic-optimized ensembles outperform both individual models and heuristically constructed ensembles. The improvements range from 1.65% (on the already high-performing Segment dataset) to 5.22% (on the more challenging Liver dataset), with larger gains observed on datasets where individual models exhibit greater diversity in error patterns. The selective ensemble approach, where GA identifies the optimal subset of models to include, is particularly effective at eliminating weak or redundant models that would otherwise degrade ensemble performance through negative transfer [116].

4.6 | Reinforcement Learning and Metaheuristic Hybrids

The intersection of metaheuristic optimization and RL has produced a growing body of work addressing policy optimization, reward shaping, and environment design through population-based ES [37], [122]. Neuroevolution—the application of evolutionary algorithms to optimize neural network architectures and weights for RL agents—has demonstrated competitive performance with gradient-based RL methods, particularly in environments with sparse rewards, deceptive gradients, or large action spaces where policy gradient methods struggle [123].

Salimans et al. [124] demonstrated that ES could achieve performance competitive with state-of-the-art RL algorithms on Atari games and MuJoCo locomotion tasks, with the advantage of linear scalability across hundreds of CPUs and no need for backpropagation through time. Such et al. [109] further showed that simple GAs applied to evolve neural network weights could match the performance of deep RL methods on several Atari games, challenging the assumption that gradient information is necessary for effective policy optimization. More recently, Covariance Matrix Adaptation Evolution Strategy (CMA-ES) has been applied to optimize reward shaping functions, automatically discovering auxiliary reward signals that accelerate RL training by 40%–60% in sparse-reward environments [125]. Population-Based Training (PBT), introduced by Jaderberg et al. [126], combines evolutionary selection with gradient-based training by maintaining a population of agents whose hyperparameters (learning rate, entropy regularization, discount factor) are mutated and selected based on performance, effectively performing online HPO for RL agents.

5 | Comparative Performance Analysis

5.1 | Cross-Domain Algorithm Benchmarking

To systematically assess the relative effectiveness of different metaheuristic algorithms across AI optimization tasks, we synthesized results from the reviewed studies using standardized comparison metrics. For each AI task–dataset combination, we identified the best-performing and second-best-performing metaheuristic algorithm, computed the performance gap, and assessed statistical significance using reported p-values from Wilcoxon signed-rank tests or Friedman tests [127]. *Table 8* presents the cross-domain benchmarking results.

Table 8. Cross-domain metaheuristic benchmarking across AI optimization tasks.

AI Task	Problem Dimension	Best Metaheuristic	Second Best	Performance Gap (%)	p-value	Friedman Rank
CNN NAS	~10^{15}	GA	PSO	0.34	0.042	1.83
RNN/LSTM NAS	~10^{8}	DE	GWO	0.52	0.031	1.67
Transformer NAS	~10^{12}	GA	WOA	0.41	0.068	2.08
SVM HPO	2–5	PSO	GWO	0.87	0.018	1.42
CNN HPO	8–15	DE	PSO	0.63	0.035	1.58
XGBoost HPO	6–12	DE	GA	0.45	0.047	1.75
Low-Dim feature selection	20–60	GWO	WOA	1.24	0.008	1.33
High-Dim feature selection	500–7,000+	HHO	MPA	0.92	0.012	1.50
Weight initialization	10^{6}–10^{8}	PSO	SA	0.28	0.091	2.17
Ensemble optimization	10–50	GA	DE	0.71	0.023	1.58

The cross-domain analysis reveals several algorithm-specific strengths. GA and DE dominate NAS tasks, likely due to their effective handling of discrete, variable-length representations and their ability to maintain population diversity across complex architectural search spaces. PSO and DE excel in continuous HPO tasks, benefiting from their efficient traversal of smooth, continuous hyperparameter landscapes. For feature selection, GWO dominates low-dimensional problems while HHO shows superior performance on high-dimensional datasets—a finding consistent with HHO's progressive transition from exploration to exploitation that is well-suited for navigating vast binary search spaces [28]. Notably, the performance gaps between the best and second-best algorithms are generally modest (0.28–1.24%), and statistical significance is not always achieved ($p > 0.05$ for transformer NAS and weight initialization), suggesting that algorithm selection is less critical than careful implementation and parameter tuning [128].

5.2|Metaheuristics vs. Bayesian Optimization vs. Random Search

A central question in the AI optimization literature is whether metaheuristic algorithms offer genuine advantages over simpler alternatives, particularly random search [34] and BO [35]. Bergstra and Bengio [34] demonstrated that random search is surprisingly competitive for HPO due to the low effective dimensionality of many hyperparameter spaces, while BO, leveraging GP surrogates, achieves superior sample efficiency on low-dimensional problems [129]. We synthesized comparative results from 42 studies that included both metaheuristic and non-metaheuristic baselines. *Table 9* presents the aggregated comparison.

Table 9. Comparison of metaheuristics, BO, and random search across AI tasks.

Method	NAS Accuracy (%)	HPO Accuracy (%)	Feature Sel. Accuracy (%)	Avg. Evaluations	Avg. Time (h)
Random Search	94.82	91.34	83.67	1,000	28.4
GP	96.14	93.87	86.41	150	12.6
Bayesian Opt. (TPE)	96.28	94.12	87.23	200	14.2
Best Metaheuristic (overall)	97.08	94.56	91.82	350	18.7
Hybrid Meta. + Surrogate	97.32	94.78	92.14	180	10.3

The aggregated comparison reveals a performance hierarchy that varies by task complexity. For NAS—the most complex and high-dimensional task—metaheuristic algorithms achieve the highest accuracy (97.08%), outperforming BO by approximately 0.8–0.9 percentage points. This advantage is attributed to the population-based nature of metaheuristics, which enables parallel exploration of diverse architectural regions, and their ability to handle the mixed-type, conditional search spaces characteristic of NAS without requiring differentiability or surrogate approximations [45]. For HPO, the advantage of metaheuristics over BO narrows to 0.4–0.7 percentage points, reflecting the effectiveness of GP surrogates for the lower-dimensional, predominantly continuous hyperparameter spaces [130]. For feature selection, metaheuristics demonstrate the largest advantage (4.6–5.4 percentage points over BO), as the binary, combinatorial nature of feature selection is poorly suited to the continuous surrogate models underlying BO [131].

Hybrid approaches combining metaheuristic search with surrogate models achieve the best overall performance across all tasks while simultaneously reducing the number of required evaluations by approximately 49% compared to standalone metaheuristics. This finding suggests that the most effective strategy is not to choose between metaheuristics and BO, but to integrate them in complementary roles—using the surrogate model to cheaply pre-screen candidate solutions and the metaheuristic to guide the global search strategy [132].

5.3 | Hybrid Metaheuristic–Artificial Intelligence Approaches

The combination of metaheuristic algorithms with specific AI models has produced a class of hybrid approaches that leverage the strengths of both components synergistically. These hybrids typically use the metaheuristic to optimize the AI model's architecture or hyperparameters while the AI model serves as the fitness evaluator or, in some cases, as a surrogate for fitness estimation. *Table 10* summarizes representative hybrid approaches and their reported performance improvements.

Table 10. Performance of hybrid metaheuristic–AI approaches.

Hybrid Method	Application	Accuracy Improvement (%)	Computation Reduction (%)	Key Reference
PSO-CNN	Image classification (CIFAR-10)	+2.34	−45	[80]
GA-LSTM	Time series forecasting	+3.17	−38	[133]
DE-Transformer	NLP text classification	+1.82	−52	[134]
GWO-SVM	Medical diagnosis	+4.56	−61	[78]
WOA-DenseNet	Object detection	+1.93	−40	[112]
HHO-XGBoost	Fraud detection	+3.42	−55	[135]
ABC-RF	Remote sensing classification	+2.78	−48	[136]
MPA-EfficientNet	Medical image analysis	+1.64	−43	[137]

The hybrid approaches consistently demonstrate both accuracy improvements (1.64–4.56%) and computation reductions (38–61%) compared to default configurations or manual tuning. The largest accuracy improvements are observed for classical ML models (GWO-SVM: +4.56%) where the hyperparameter space is relatively small but the sensitivity to parameter settings is high. Computation reductions are most pronounced for hybrid approaches that incorporate early stopping and surrogate-assisted evaluation, preventing the waste of resources on unpromising configurations [138].

5.4 | Statistical Analysis and Critical Difference Assessment

To rigorously assess the statistical significance of performance differences among metaheuristic algorithms across AI tasks, we employed the Friedman test followed by Nemenyi post-hoc analysis, following the methodology recommended by Demšar [127] for comparing multiple classifiers over multiple datasets. The Friedman test rejected the null hypothesis of equal algorithm performance across all tasks ($\chi^2 = 47.83$, $p < 0.001$, $df = 9$), confirming significant differences among the tested algorithms.

The Nemenyi post-hoc analysis with a Critical Difference (CD) of 2.14 at $\alpha = 0.05$ revealed that: 1) DE and GA are statistically tied for overall best performance (average ranks of 2.31 and 2.48, respectively) and both significantly outperform random search (rank 8.67), 2) PSO (rank 3.12) and GWO (rank 3.45) form a second tier that is not significantly different from the top tier but significantly outperforms SA (rank 6.83) and BA (rank 7.14), and 3) recently proposed algorithms (HHO, MPA, SSA) achieve intermediate ranks (4.21–5.38) that are not significantly different from either the top or bottom tiers, suggesting that their advantages are task-specific rather than universal. These findings caution against claims of universal superiority for any single metaheuristic algorithm and support the recommendation that algorithm selection should be guided by task characteristics—problem dimensionality, variable types, evaluation cost, and time budget [128], [139].

6 | Bibliometric and Trend Analysis

The bibliometric analysis of the 347 included studies reveals an exponential growth trajectory in research at the intersection of metaheuristic algorithms and AI optimization. The annual publication count has increased from 12 papers in 2015 to 68 papers in 2024, representing a Compound Annual Growth Rate (CAGR) of approximately 21.3%. This growth substantially outpaces the general increase in AI/ML publications (CAGR

~15%) during the same period, indicating intensifying research interest in metaheuristic-driven AI optimization specifically [140]. The most dramatic growth occurred between 2019 and 2023, coinciding with the emergence of several influential new metaheuristic algorithms (HHO in 2019, MPA in 2020) and the maturation of NAS as a mainstream research topic [141].

Geographically, the research output is dominated by China (34.6% of included studies), followed by India (12.4%), Iran (8.9%), the United States (7.8%), Australia (6.3%), and Malaysia (5.2%). Institutionally, the Chinese Academy of Sciences, Huazhong University of Science and Technology, and Griffith University (Australia) are the most productive institutions, each contributing more than 10 studies to the corpus. The geographic distribution reflects the broader pattern in computational intelligence research, with Asian institutions playing an increasingly dominant role [142].

Table 11 presents the ten most active journals for metaheuristic–AI optimization research, along with their 2024 Journal Impact Factors and the number of included studies from each.

Table 11. Top 10 journals publishing metaheuristic–AI optimization research.

Rank	Journal	Publisher	Impact Factor (2024)	No. of Studies
1	Expert systems with applications	Elsevier	7.5	48
2	Applied soft computing	Elsevier	7.2	42
3	Knowledge-based systems	Elsevier	7.2	36
4	Neurocomputing	Elsevier	5.5	31
5	Neural networks	Elsevier	6.0	27
6	IEEE trans. on neural networks and learning systems	IEEE	10.2	24
7	Information sciences	Elsevier	8.1	22
8	Swarm and evolutionary computation	Elsevier	8.2	20
9	IEEE trans. on evolutionary computation	IEEE	11.7	18
10	Engineering applications of AI	Elsevier	7.5	16

The journal distribution reveals a concentration in applied soft computing and AI journals, with Elsevier journals dominating the top 10. IEEE Transactions on Evolutionary Computation, despite ranking 9th in volume, has the highest impact factor (11.7), reflecting its status as the premier outlet for theoretically rigorous evolutionary computation research. The strong representation of application-oriented journals (expert systems with applications, engineering applications of AI) indicates that the field is driven primarily by practical applications rather than purely theoretical contributions [143].

Thematic analysis of research trends reveals several evolving hot topics. Between 2015 and 2018, the dominant focus was on feature selection using swarm intelligence algorithms (PSO, GWO, WOA), accounting for approximately 45% of publications. From 2019 to 2021, NAS emerged as the fastest-growing topic, driven by the success of evolutionary NAS methods and the availability of NAS benchmark datasets (NAS-Bench-101, NAS-Bench-201). Since 2022, the field has increasingly focused on three emerging topics: 1) AutoML pipelines integrating metaheuristic optimization of feature selection, model selection, and HPO into unified frameworks, 2) explainable AI optimization, where metaheuristic decisions are made interpretable through visualization and sensitivity analysis, and 3) efficient/green NAS, where multi-objective metaheuristics simultaneously optimize accuracy, parameter count, and energy consumption [144], [145].

7 | Challenges and Limitations

Despite the demonstrated effectiveness of metaheuristic algorithms in AI optimization, several significant challenges and limitations remain that constrain their practical impact and broader adoption.

High computational cost of fitness evaluation: the most fundamental challenge is that each fitness evaluation in AI optimization requires training a ML model—a process that is orders of magnitude more expensive than evaluating classical benchmark functions [56]. A single evaluation may consume minutes to days of GPU

time, and population-based metaheuristics require hundreds to thousands of evaluations per run. While surrogate-assisted methods [59] and early stopping strategies [58] have substantially reduced this cost, the computational burden remains prohibitive for large-scale models (e.g., training a modern transformer on ImageNet requires ~300 GPU-hours per evaluation), effectively limiting metaheuristic optimization to smaller proxy tasks and datasets [146].

Scalability to very large search spaces: modern AI architectures present search spaces of staggering dimensionality. A transformer-based language model may have hundreds of billions of possible architecture configurations when considering layer count, attention head count, embedding dimensions, feed-forward widths, and activation functions at each layer [147]. Current metaheuristic NAS methods have been validated primarily on modestly sized search spaces (10^6–10^{15} configurations) and small datasets (CIFAR-10, CIFAR-100), with limited evidence of scalability to ImageNet-scale or larger problems [148].

Transferability of optimized configurations: architectures and hyperparameters discovered through metaheuristic optimization on one dataset do not necessarily transfer to other datasets or tasks [149]. This transferability gap is particularly concerning because the high cost of metaheuristic optimization makes re-optimization for each new task impractical. While some transfer learning approaches for NAS have been proposed [150], the general transferability of metaheuristic-optimized AI configurations remains an open problem.

Reproducibility concerns: the stochastic nature of metaheuristic algorithms introduces variability in results across independent runs, complicating reproducibility and fair comparison [151]. Many studies in the reviewed corpus report results from a single run or a small number of runs (3–5), without adequate statistical analysis of the distribution of outcomes. Furthermore, hardware differences (GPU type, memory capacity, parallelization strategy) can significantly affect both the search trajectory and the reported computational cost, making cross-study comparisons unreliable [152].

Lack of theoretical convergence guarantees: while classical optimization theory provides convergence guarantees for gradient-based methods under certain conditions (convexity, Lipschitz continuity), no such guarantees exist for metaheuristic algorithms applied to the non-convex, stochastic, and discontinuous fitness landscapes characteristic of AI optimization [153]. The No Free Lunch theorems further establish that no single optimization algorithm can dominate all others across all possible problems [139], implying that the observed superiority of certain metaheuristics in specific AI tasks may not generalize to other problem instances or domains.

Benchmark bias: the overwhelming reliance on a small set of benchmark datasets—particularly CIFAR-10 for NAS and UCI datasets for feature selection—raises concerns about the generalizability of reported results [154]. Models and search strategies optimized for CIFAR-10's 32×32 pixel images may not translate to higher-resolution, more complex visual recognition tasks. The AI optimization community would benefit from the development and adoption of more diverse, realistic, and challenging benchmark suites that better represent real-world deployment scenarios [155].

8 | Future Research Directions

Based on the comprehensive analysis presented in this review, we identify eight promising future research directions that have the potential to significantly advance the field of metaheuristic-driven AI optimization.

Zero-shot and few-shot NAS with metaheuristics: the development of metaheuristic NAS methods that can discover effective architectures with zero or very few full training evaluations—leveraging architecture performance predictors, training-free metrics (e.g., neural tangent kernel analysis, linear regions), and meta-learned surrogates—represents a transformative opportunity to reduce the computational cost of architecture search by orders of magnitude [156], [157]. Preliminary results suggest that training-free NAS can evaluate candidate architectures in seconds rather than hours, enabling metaheuristic search over millions of architectures within practical time budgets.

Metaheuristic optimization for foundation models and LLMs: the emergence of Large Language Models (LLMs) with billions of parameters presents both a tremendous opportunity and a formidable challenge for metaheuristic optimization [158]. Key optimization targets include attention pattern design, mixture-of-experts routing, prompt template optimization, and adapter/LoRA configuration for efficient fine-tuning. The scale of these models demands new metaheuristic strategies that can operate in extremely high-dimensional spaces with minimal fitness evaluations [159].

Federated metaheuristic optimization for privacy-preserving AI: in privacy-sensitive domains (healthcare, finance), training data cannot be centralized. Federated metaheuristic optimization—where each participant optimizes a local model population and shares only aggregate fitness information—could enable collaborative architecture and hyperparameter search without exposing private data [160]. The design of communication-efficient, privacy-preserving population exchange protocols for federated settings is an open research challenge.

Green AI: energy-efficient architecture search. As the environmental cost of AI training attracts increasing scrutiny, multi-objective metaheuristic optimization that simultaneously considers accuracy, parameter count, Floating Point Operations (FLOPs), memory footprint, and energy consumption becomes essential [161]. Pareto-based metaheuristics (NSGA-II, MOEA/D) are naturally suited to this multi-objective formulation and can discover architectures that achieve competitive accuracy at a fraction of the computational and environmental cost.

Quantum-inspired metaheuristics for quantum ML: quantum computing offers exponential speedups for certain optimization problems, and quantum-inspired metaheuristic algorithms that incorporate quantum mechanical principles (superposition, entanglement, quantum walks) into classical search procedures have shown promising results on combinatorial optimization benchmarks [162]. Applying these quantum-inspired methods to optimize quantum neural network circuits and variational quantum algorithms represents a nascent but potentially transformative research direction.

Self-adaptive metaheuristics with online learning: current metaheuristic algorithms typically use fixed or predetermined parameter adaptation schedules. Integrating RL or bandit-based online learning into the metaheuristic search process—enabling the algorithm to learn which operators, parameter settings, and search strategies are most effective for the current problem instance in real time—could substantially improve algorithm robustness and eliminate the need for manual algorithm tuning [163], [164].

Explainable metaheuristic decisions in AI model design: as AI systems are deployed in safety-critical applications, understanding why a metaheuristic selected a particular architecture or hyperparameter configuration becomes important. Developing interpretability tools that analyze the search trajectory, identify influential genes/dimensions, and provide human-understandable explanations for optimization decisions is an emerging research need [165].

Unified AutoML frameworks with metaheuristic optimization: the integration of metaheuristic optimization across the entire ML pipeline—data preprocessing, feature engineering, model selection, architecture search, hyperparameter tuning, and ensemble construction—within unified, user-friendly AutoML frameworks represents the ultimate practical goal. While existing frameworks (Auto-WEKA, Auto-sklearn, FLAML) primarily use BO, incorporating population-based metaheuristic search could improve performance on complex, large-scale ML pipeline optimization problems [166], [167].

9 | Conclusion

This comprehensive systematic review has examined 347 peer-reviewed studies published between 2015 and 2025, providing a rigorous and structured analysis of metaheuristic optimization algorithms applied to AI model design and configuration. The review has addressed three interconnected RQs concerning the effectiveness, comparative advantages, and future trajectory of metaheuristic-driven AI optimization across NAS, HPO, feature selection, training optimization, ensemble learning, and RL.

The evidence synthesized in this review leads to several key conclusions:

I. Metaheuristic algorithms have demonstrated consistent and statistically significant effectiveness across all examined AI optimization tasks, with evolutionary algorithms (GA, DE) excelling in discrete architectural search spaces and swarm intelligence methods (PSO, GWO, HHO) showing particular strength in continuous hyperparameter tuning and binary feature selection. Average accuracy improvements of 2.3–5.8% over default configurations, combined with feature reduction ratios exceeding 90% in high-dimensional datasets, confirm the practical value of metaheuristic approaches for AI model optimization.

II. The comparative analysis reveals that metaheuristic algorithms outperform random search and BO in complex, high-dimensional, and combinatorial AI optimization scenarios—particularly NAS and feature selection—while hybrid approaches combining metaheuristic search with surrogate models achieve the best overall performance across all tasks. The findings support a pragmatic strategy of algorithm selection based on task characteristics rather than allegiance to any single method.

III. Despite remarkable progress, the field faces significant challenges including computational scalability, reproducibility, transferability of optimized configurations, and the absence of theoretical convergence guarantees. The identified future research directions—zero-shot NAS, LLM optimization, federated metaheuristic search, green AI, quantum-inspired methods, self-adaptive algorithms, explainable optimization, and unified AutoML frameworks—outline a rich agenda for advancing the field toward greater practical impact and theoretical maturity.

As AI systems continue to grow in complexity and are deployed across an ever-expanding range of applications, the demand for automated, efficient, and robust optimization methods will only intensify. Metaheuristic algorithms, with their flexibility, population-based parallelism, and proven effectiveness across diverse optimization landscapes, are uniquely positioned to meet this demand. The continued co-evolution of metaheuristic intelligence and AI promises to yield increasingly powerful, efficient, and accessible AI systems that push the boundaries of what ML can achieve.

Author Contribution

A. R. K. conceived the study, conducted the literature search, and wrote the manuscript. L. M. C. performed data extraction, bibliometric analysis, and statistical analysis. D. A. F. contributed to the quality assessment, comparative analysis, and critical revision. All authors reviewed and approved the final manuscript.

Data Availability

All data supporting the findings of this review are available within the article. The complete dataset of extracted study records is available from the corresponding author upon reasonable request.

Funding

This research received no external funding.

Conflicts of Interest

The authors declare no conflict of interest.

Consent for Publication

All authors have provided their consent for the publication of this manuscript.

Ethics Approval and Consent to Participate

This article does not involve studies with human participants or animals conducted by any authors.

References

[1] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

[2] Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, *349*(6245), 255–260. https://doi.org/10.1126/science.aaa8415

[3] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press Cambridge. https://mitpress.mit.edu/9780262035613/deep-learning/

[4] Talbi, E. G. (2009). *Metaheuristics: From design to implementation*. Wiley. https://www.wiley.com/en-us/Metaheuristics%3A+From+Design+to+Implementation+-p-9780470278581

[5] Yang, X. S. (2020). *Nature-inspired optimization algorithms*. Elsevier. https://shop.elsevier.com/books/nature-inspired-optimization-algorithms/yang/978-0-12-821986-7

[6] Boussaïd, I., Lepagnot, J., & Siarry, P. (2013). A survey on optimization metaheuristics. *Information sciences*, *237*, 82–117. https://doi.org/10.1016/j.ins.2013.02.041

[7] Karimi-Mamaghan, M., Mohammadi, M., Meyer, P., Karimi-Mamaghan, A. M., & Talbi, E. G. (2022). Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: A state-of-the-art. *European journal of operational research*, *296*(2), 393–422. https://doi.org/10.1016/j.ejor.2021.04.032

[8] Handoko, S. D., Nguyen, D. T., Yuan, Z., & Lau, H. (2014). Reinforcement learning for adaptive operator selection in memetic search applied to quadratic assignment problem. *GECCO comp '14: Proceedings of the companion publication of the 2014 annual conference on genetic and evolutionary computation* (pp. 193–194). ACM Digital Library. https://doi.org/10.1145/2598394.2598451

[9] Dokeroglu, T., Canturk, D., & Kucukyilmaz, T. (2024). *A survey on pioneering metaheuristic algorithms between 2019 and 2024*. https://doi.org/10.48550/arXiv.2501.14769

[10] Zoph, B., & Le, Q. V. (2016). *Neural architecture search with reinforcement learning*. https://doi.org/10.48550/arXiv.1611.01578

[11] Elsken, T., Metzen, J. H., & Hutter, F. (2019). Neural architecture search: A survey. *Journal of machine learning research*, *20*(55), 1–21. http://jmlr.org/papers/v20/18-598.html

[12] Feurer, M., & Hutter, F. (2019). Hyperparameter optimization. In *Automated machine learning* (pp. 3–33). Springer, Cham. https://doi.org/10.1007/978-3-030-05318-5_1

[13] Yu, T., & Zhu, H. (2020). *Hyper-parameter optimization: A review of algorithms and applications*. https://doi.org/10.48550/arXiv.2003.05689

[14] Xue, B., Zhang, M., Browne, W. N., & Yao, X. (2016). A survey on evolutionary computation approaches to feature selection. *IEEE transactions on evolutionary computation*, *20*(4), 606–626. https://doi.org/10.1109/TEVC.2015.2504420

[15] Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2017). Feature selection: A data perspective. *ACM computing surveys (CSUR)*, *50*(6), 1–45. https://doi.org/10.1145/3136625

[16] Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (pp. 249-256). JMLR Workshop and Conference Proceedings. https://proceedings.mlr.press/v9/glorot10a/glorot10a.pdf

[17] Loshchilov, I., & Hutter, F. (2017). SGDR: Stochastic gradient descent with warm restarts. *5th international conference on learning representations (ICLR 2017)* (PP. 1-11). OpenReview. https://researchr.org/publication/LoshchilovH17

[18] Li, H., Xu, Z., Taylor, G., Studer, C., & Goldstein, T. (2018). Visualizing the loss landscape of neural nets. *Advances in neural information processing systems 31 (NeurIPS 2018)* (pp. 6389–6399). Curran Associates, Inc. https://proceedings.neurips.cc/paper/2018/hash/a41b3bb3e6b050b6c9067c67f663b915-Abstract.html

[19] Zhou, Z. H. (2012). *Ensemble methods: Foundations and algorithms*. CRC Press. https://doi.org/10.1201/b12207

[20] Holand, J. H. (1975). *Adaptation in natural and artificial systems*. The MIT Press. https://mitpress.mit.edu/9780262082136/adaptation-in-natural-and-artificial-systems/

[21] Storn, R., & Price, K. (1997). Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, *11*(4), 341–359. https://doi.org/10.1023/A:1008202821328

[22] Beyer, H. G., & Schwefel, H. P. (2002). Evolution strategies – A comprehensive introduction. *Natural computing*, *1*(1), 3–52. https://doi.org/10.1023/A:1015059928466

[23] Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95 - international conference on neural networks* (pp. 1942–1948). IEEE. https://doi.org/10.1109/ICNN.1995.488968

[24] Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: Optimization by a colony of cooperating agents. *IEEE transactions on systems, man, and cybernetics, part b (cybernetics)*, *26*(1), 29–41. https://doi.org/10.1109/3477.484436

[25] Karaboga, D. (2005). *An idea based on honey bee swarm for numerical optimization*. https://abc.erciyes.edu.tr/pub/tr06_2005.pdf

[26] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software*, *69*, 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007

[27] Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in engineering software*, *95*, 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

[28] Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future generation computer systems*, *97*, 849–872. https://doi.org/10.1016/j.future.2019.02.028

[29] Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (2017). Salp swarm algorithm: A bio-inspired optimizer for engineering design problems. *Advances in engineering software*, *114*, 163–191. https://doi.org/10.1016/j.advengsoft.2017.07.002

[30] Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*(4598), 671–680. https://doi.org/10.1126/science.220.4598.671

[31] Mirjalili, S. (2016). SCA: A sine cosine algorithm for solving optimization problems. *Knowledge-based systems*, *96*, 120–133. https://doi.org/10.1016/j.knosys.2015.12.022

[32] Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-based systems*, *89*, 228–249. https://doi.org/10.1016/j.knosys.2015.07.006

[33] Osaba, E., Del Ser, J., Sadollah, A., Bilbao, M. N., & Camacho, D. (2018). A discrete water cycle algorithm for solving the symmetric and asymmetric traveling salesman problem. *Applied soft computing*, *71*, 277–290. https://doi.org/10.1016/j.asoc.2018.06.047

[34] Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of machine learning research*, *13*(2), 281–305. http://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf

[35] Snoek, J., Larochelle, H., & Adams, R. (2012). Practical Bayesian optimization of machine learning algorithms. *Advances in neural information processing systems* (Vol. 25, pp. 2951–2959). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2012/file/05311655a15b75fab86956663e1819cd-Paper.pdf

[36] Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2011). Sequential model-based optimization for general algorithm configuration. In *Learning and intelligent optimization* (pp. 507–523). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-25566-3_40

[37] Miikkulainen, R., & Forrest, S. (2021). A biological perspective on evolutionary computation. *Nature machine intelligence*, *3*(1), 9–15. https://doi.org/10.1038/s42256-020-00278-8

[38] Fister Jr, I., Yang, X.-S., Fister, I., Brest, J., & Fister, D. (2013). A brief review of nature-inspired algorithms for optimization. *Elektrotehniski vestnik*, *80*(3), 116–122. https://www.researchgate.net/publication/249645112

[39] Abdel-Basset, M., Abdel-Fatah, L., & Sangaiah, A. K. (2018). Metaheuristic algorithms: A comprehensive review. In *Computational intelligence for multimedia big data on the cloud with engineering applications* (pp. 185–231). Academic Press. https://doi.org/10.1016/B978-0-12-813314-9.00010-4

[40] Goldberg, D. E. (1989). *Genetic algorithms in search, optimization & machine learning*. Addison-Wesley. https://www.amazon.fr/Algorithms-Optimization-Learning-Goldberg-published/dp/B00E31KI3G

[41] Xie, L., & Yuille, A. (2017). Genetic CNN. *Proceedings of the IEEE international conference on computer vision (ICCV 2017)* (pp. 1379–1388). IEEE. https://doi.org/10.1109/ICCV.2017.154

[42] Das, S., & Suganthan, P. N. (2011). Differential evolution: A survey of the state-of-the-art. *IEEE transactions on evolutionary computation*, *15*(1), 4–31. https://doi.org/10.1109/TEVC.2010.2059031

[43] Mafarja, M., Aljarah, I., Faris, H., Hammouri, A. I., Al-Zoubi, A. M., & Mirjalili, S. (2019). Binary grasshopper optimisation algorithm approaches for feature selection problems. *Expert systems with applications*, *117*, 267–286. https://doi.org/10.1016/j.eswa.2018.09.015

[44] Bischl, B., Richter, J., Becker, M., Binder, M., & Pielok, T. (2023). Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. *WIREs data mining and knowledge discovery*, *13*(2), 1–43. https://doi.org/10.1002/widm.1484

[45] White, C., Neiswanger, W., & Savani, Y. (2021). BANANAS: Bayesian optimization with neural architectures for neural architecture search. *Proceedings of the AAAI conference on artificial intelligence*, (Vol. 35, No. 12, PP. 10293-10301). https://doi.org/10.1609/aaai.v35i12.17233

[46] Liu, H., Simonyan, K., & Yang, Y. (2018). *Darts: Differentiable architecture search.* https://doi.org/10.48550/arXiv.1806.09055

[47] Pham, H., Guan, M., Zoph, B., Le, Q., & Dean, J. (2018). Efficient neural architecture search via parameters sharing. *Proceedings of the 35th international conference on machine learning* (pp. 4095–4104). Proceedings of Machine Learning Research (PMLR). https://proceedings.mlr.press/v80/pham18a.html

[48] Zoph, B., Vasudevan, V., Shlens, J., & Le, Q. V. (2018). Learning transferable architectures for scalable image recognition. *2018 IEEE/CVF conference on computer vision and pattern recognition (CVPR)* (pp. 8697–8710). IEEE. https://doi.org/10.1109/CVPR.2018.00907

[49] Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of machine learning research*, *3*, 1157–1182. https://www.jmlr.org/papers/v3/guyon03a.html

[50] Eiben, A. E., & Smith, J. E. (2015). *Introduction to evolutionary computing*. Springer Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-44874-8

[51] Črepinšek, M., Liu, S. H., & Mernik, M. (2013). Exploration and exploitation in evolutionary algorithms: A survey. *ACM computing surveys (CSUR)*, *45*(3), 1–33. https://doi.org/10.1145/2480741.2480752

[52] Alba, E., & Dorronsoro, B. (2008). *Cellular genetic algorithms*. Springer New York, NY. https://doi.org/10.1007/978-0-387-77610-1

[53] Morales-Castañeda, B., Zaldívar, D., Cuevas, E., Fausto, F., & Rodríguez, A. (2020). A better balance in metaheuristic algorithms: Does it exist? *Swarm and evolutionary computation*, *54*, 100671. https://doi.org/10.1016/j.swevo.2020.100671

[54] White, C., Zela, A., Ru, R., Liu, Y., & Hutter, F. (2021). How powerful are performance predictors in neural architecture search? *Advances in neural information processing systems* (pp. 28454–28469). Curran Associates, Inc. https://proceedings.neurips.cc/paper/2021/hash/ef575e8837d065a1683c022d2077d342-Abstract.html

[55] Karafotias, G., Hoogendoorn, M., & Eiben, A. E. (2015). Parameter control in evolutionary algorithms: Trends and challenges. *IEEE transactions on evolutionary computation*, *19*(2), 167–187. https://doi.org/10.1109/TEVC.2014.2308294

[56] Salmani Pour Avval, S., Eskue, N. D., Groves, R. M., & Yaghoubi, V. (2025). Systematic review on neural architecture search. *Artificial intelligence review*, *58*(3), 73. https://doi.org/10.1007/s10462-024-11058-w

[57] Baker, B., Gupta, O., Raskar, R., & Naik, N. (2017). *Accelerating neural architecture search using performance prediction*. https://doi.org/10.48550/arXiv.1705.10823

[58] Li, L., & Talwalkar, A. (2020). Random search and reproducibility for neural architecture search. *Proceedings of the 35th uncertainty in artificial intelligence conference* (pp. 367–377). Proceedings of Machine Learning Research (PMLR). https://proceedings.mlr.press/v115/li20c.html

[59] Liu, Y., Sun, Y., Xue, B., Zhang, M., Yen, G. G., & Tan, K. C. (2023). A survey on evolutionary neural architecture search. *IEEE transactions on neural networks and learning systems*, *34*(2), 550–570. https://doi.org/10.1109/TNNLS.2021.3100554

[60] Real, E., Aggarwal, A., Huang, Y., & Le, Q. V. (2019). Regularized evolution for image classifier architecture search. *Proceedings of the AAAI conference on artificial intelligence* (pp. 4780–4789). AAAI Press. https://doi.org/10.1609/aaai.v33i01.33014780

[61] Ren, P., Xiao, Y., Chang, X., Huang, P. Y., Li, Z., Chen, X., & Wang, X. (2021). A comprehensive survey of neural architecture search: Challenges and solutions. *ACM computing surveys (CSUR)*, *54*(4), 1–34. https://doi.org/10.1145/3447582

[62] Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., … ., & Moher, D. (2021). The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *British medical journal*, *372*. https://doi.org/10.1136/bmj.n71

[63] Real, E., Liang, C., So, D., & Le, Q. (2020). AutoML-zero: Evolving machine learning algorithms from scratch. *Proceedings of the 37th international conference on machine learning* (pp. 8007–8019). Proceedings of Machine Learning Research (PMLR). https://proceedings.mlr.press/v119/real20a.html

[64] Sun, Y., Xue, B., Zhang, M., & Yen, G. G. (2020). Evolving deep convolutional neural networks for image classification. *IEEE transactions on evolutionary computation*, *24*(2), 394–407. https://doi.org/10.1109/TEVC.2019.2916183

[65] Liu, H., Simonyan, K., Vinyals, O., Fernando, C., & Kavukcuoglu, K. (2017). *Hierarchical representations for efficient architecture search*. https://doi.org/10.48550/arXiv.1711.00436

[66] Junior, F. E. F., & Yen, G. G. (2019). Particle swarm optimization of deep neural networks architectures for image classification. *Swarm and evolutionary computation*, *49*, 62–74. https://doi.org/10.1016/j.swevo.2019.05.010

[67] Brodzicki, A., Piekarski, M., & Jaworek-Korjakowska, J. (2021). The whale optimization algorithm approach for deep neural networks. *Sensors, 21*(23), 8003. https://doi.org/10.3390/s21238003

[68] Singh, T., Solanki, A., Sharma, S. K., Jhanjhi, N. Z., & Ghoniem, R. M. (2023). Grey wolf optimization-based CNN-LSTM network for the prediction of energy consumption in smart home environment. *IEEE access*, *11*, 114917-114935. https://doi.org/10.1109/ACCESS.2023.3311751

[69] Lu, Z., Whalen, I., Boddeti, V., Dhebar, Y., Deb, K., Goodman, E., & Banzhaf, W. (2019). NSGA-net: Neural architecture search using multi-objective genetic algorithm. *Proceedings of the genetic and evolutionary computation conference* (pp. 419–427). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3321707.3321729

[70] Lu, Z., Whalen, I., Dhebar, Y., Deb, K., Goodman, E. D., Banzhaf, W., & Boddeti, V. N. (2021). Multiobjective evolutionary design of deep convolutional neural networks for image classification. *IEEE transactions on evolutionary computation*, *25*(2), 277–291. https://doi.org/10.1109/TEVC.2020.3024708

[71] Gu, H., Wang, H., & Jin, Y. (2022). Surrogate-assisted differential evolution with adaptive multi-subspace search for large-scale expensive optimization. *IEEE transactions on evolutionary computation*, 27(6), 1765 - 1779. https://doi.org/10.1109/TEVC.2022.3226837

[72] Ghosh, A., Jana, N. D., & Ghosh, S. (2025). Automated CNN architecture design with enhanced particle swarm optimization. *Journal of heuristics*, *31*(4), 35. https://doi.org/10.1007/s10732-025-09570-5

[73] Faramarzi, A., Heidarinejad, M., Mirjalili, S., & Gandomi, A. H. (2020). Marine predators algorithm: A nature-inspired metaheuristic. *Expert systems with applications*, *152*, 113377. https://doi.org/10.1016/j.eswa.2020.113377

[74] Franceschi, L., Donini, M., Perrone, V., Klein, A., Archambeau, C., Seeger, M., … ., & Frasconi, P. (2025). Hyperparameter optimization in machine learning. *Foundations and trends in machine learning*, *18*(6), 975–1109. https://doi.org/10.1561/2200000088

[75] Lorenzo, P. R., Nalepa, J., Kawulok, M., Ramos, L. S., & Pastor, J. R. (2017). Particle swarm optimization for hyper-parameter selection in deep neural networks. *Proceedings of the genetic and evolutionary computation conference* (pp. 481–488). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3071178.3071208

[76] Xue, B., Zhang, M., & Browne, W. N. (2014). Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms. *Applied soft computing*, *18*, 261–276. https://doi.org/10.1016/j.asoc.2013.09.018

[77] Ibrahim, M. Q., Hussein, N. K., Guinovart, D., & Qaraad, M. (2025). Optimizing convolutional neural networks: A comprehensive review of hyperparameter tuning through metaheuristic algorithms. *Archives of computational methods in engineering*, *32*(8), 5123–5160. https://doi.org/10.1007/s11831-025-10292-x

[78] Emary, E., Zawbaa, H. M., & Hassanien, A. E. (2016). Binary grey wolf optimization approaches for feature selection. *Neurocomputing*, *172*, 371–381. https://doi.org/10.1016/j.neucom.2015.06.083

[79] Albelwi, S., & Mahmood, A. (2017). A framework for designing the architectures of deep convolutional neural networks. *Entropy*, *19*(6), 1–20. https://doi.org/10.3390/e19060242

[80] Chen, K., & Xie, J. (2025). Hybrid adaptive Wolf-Particle swarm optimization algorithm and its application in CNN neural network hyperparameters optimization. *Discover computing*, *28*(1), 319. https://doi.org/10.1007/s10791-025-09878-7

[81] Al-Tashi, Q., Abdulkadir, S. J., Rais, H. M., Mirjalili, S., & Alhussian, H. (2020). Binary optimization using hybrid grey wolf optimization for feature selection. *IEEE access*, 7(1), 39496-39508. https://doi.org/10.1109/ACCESS.2019.2906757

[82] Ibrahim, R. A., Elaziz, M. A., & Lu, S. (2018). Chaotic opposition-based grey-wolf optimization algorithm based on differential evolution and disruption operator for global optimization. *Expert systems with applications*, *108*, 1–27. https://doi.org/10.1016/j.eswa.2018.04.028

[83] Yang, X. S. (2010). A new metaheuristic bat-inspired algorithm. In *Nature inspired cooperative strategies for optimization (NICSO 2010)* (pp. 65–74). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-12538-6_6

[84] Coppola, C., Papa, L., Boresta, M., Amerini, I., & Palagi, L. (2024). Tuning parameters of deep neural network training algorithms pays off: A computational study. *Transactions in operations research (TOP)*, *32*(3), 579–620. https://doi.org/10.1007/s11750-024-00683-x

[85] Probst, P., Boulesteix, A. L., & Bischl, B. (2019). Tunability: Importance of hyperparameters of machine learning algorithms. *Journal of machine learning research*, *20*(53), 1–32. http://jmlr.org/papers/v20/18-444.html

[86] Golovin, D., Solnik, B., Moitra, S., Kochanski, G., Karro, J., & Sculley, D. (2017). Google vizier: A service for black-box optimization. *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining (KDD '17)* (pp. 1487–1495). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3097983.3098043

[87] Bischl, B., Casalicchio, G., Feurer, M., Gijsbers, P., Hutter, F., Lang, M., ... & Vanschoren, J. (2017). *Openml benchmarking suites.* https://doi.org/10.48550/arXiv.1708.03731

[88] Chandrashekar, G., & Sahin, F. (2014). A survey on feature selection methods. *Computers & electrical engineering*, *40*(1), 16–28. https://doi.org/10.1016/j.compeleceng.2013.11.024

[89] Kaur, A., Chhabbra, A., & Shivani. (2024). A comprehensive review of feature selection techniques with metaheuristic algorithms (2019–2024). *International conference on information and communication technology for competitive strategies* (pp. 401-417). Singapore: Springer Nature Singapore. https://doi.org/10.1007/978-981-96-4142-0_34

[90] Hussain, K., Mohd Salleh, M. N., Cheng, S., & Shi, Y. (2019). Metaheuristic research: A comprehensive survey. *Artificial intelligence review*, *52*(4), 2191–2233. https://doi.org/10.1007/s10462-017-9605-z

[91] Nguyen, B. H., Xue, B., & Zhang, M. (2020). A survey on swarm intelligence approaches to feature selection in data mining. *Swarm and evolutionary computation*, *54*, 100663. https://doi.org/10.1016/j.swevo.2020.100663

[92] Mafarja, M., & Mirjalili, S. (2018). Whale optimization approaches for wrapper feature selection. *Applied soft computing*, *62*, 441–453. https://doi.org/10.1016/j.asoc.2017.11.006

[93] Xue, B., Zhang, M., & Browne, W. N. (2013). Particle swarm optimization for feature selection in classification: A multi-objective approach. *IEEE transactions on cybernetics*, *43*(6), 1656–1671. https://doi.org/10.1109/TSMCB.2012.2227469

[94] Too, J., & Mirjalili, S. (2021). A hyper learning binary dragonfly algorithm for feature selection: A COVID-19 case study. *Knowledge-based systems*, *212*, 106553. https://doi.org/10.1016/j.knosys.2020.106553

[95] Siedlecki, W., & Sklansky, J. (1989). A note on genetic algorithms for large-scale feature selection. *Pattern recognition letters*, *10*(5), 335–347. https://doi.org/10.1016/0167-8655(89)90037-8

[96] Faris, H., Mafarja, M. M., Heidari, A. A., Aljarah, I., Al-Zoubi, A. M., Mirjalili, S., & Fujita, H. (2018). An efficient binary Salp Swarm algorithm with crossover scheme for feature selection problems. *Knowledge-based systems*, *154*, 43–67. https://doi.org/10.1016/j.knosys.2018.05.009

[97] Cui, X., Luo, Q., Zhou, Y., Deng, W., & Yin, S. (2022). Quantum-inspired moth-flame optimizer with enhanced local search strategy for cluster analysis. *Frontiers in bioengineering and biotechnology*, *10*, 908356. https://doi.org/10.3389/fbioe.2022.908356

[98] Nenavath, H., & Jatoth, R. K. (2018). Hybridizing sine Cosine algorithm with differential evolution for global optimization and object tracking. *Applied soft computing*, *62*, 1019–1043. https://doi.org/10.1016/j.asoc.2017.09.039

[99] Abd Elaziz, M., Ewees, A. A., Yousri, D., Abualigah, L., & Al-qaness, M. A. A. (2022). Modified marine predators algorithm for feature selection: Case study metabolomics. *Knowledge and information systems*, *64*(1), 261–287. https://doi.org/10.1007/s10115-021-01641-w

[100] Hancer, E., Xue, B., & Zhang, M. (2018). Differential evolution for filter feature selection based on information theory and feature ranking. *Knowledge-based systems*, *140*, 103–119. https://doi.org/10.1016/j.knosys.2017.10.028

[101] Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for Cancer classification using support vector machines. *Machine learning*, *46*(1), 389–422. https://doi.org/10.1023/A:1012487302797

[102] Mirjalili, S. (2019). Genetic algorithm. In *Evolutionary algorithms and neural networks* (pp. 43-55). Springer International Publishing. https://www.springerprofessional.de/en/genetic-algorithm/15882800

[103] Koza, J. R. (1992). *Genetic programming on the programming of computers by means of natural selection*. MIT Press. https://mitpress.mit.edu/9780262527910/genetic-programming/

[104] Tran, B., Xue, B., & Zhang, M. (2019). Variable-length particle swarm optimization for feature selection on high-dimensional classification. *IEEE transactions on evolutionary computation*, *23*(3), 473–487. https://doi.org/10.1109/TEVC.2018.2869405

[105] Ruder, S. (2016). *An overview of gradient descent optimization algorithms*. https://doi.org/10.48550/arXiv.1609.04747

[106] Ding, S., Li, H., Su, C., Yu, J., & Jin, F. (2013). Evolutionary artificial neural networks: A review. *Artificial intelligence review*, *39*(3), 251–260. https://doi.org/10.1007/s10462-011-9270-6

[107] Smith, L. N. (2017). Cyclical learning rates for training neural networks. *2017 IEEE winter conference on applications of computer vision (WACV)* (pp. 464–472). IEEE. https://doi.org/10.1109/WACV.2017.58

[108] Liang, J., Meyerson, E., Hodjat, B., Fink, D., Mutch, K., & Miikkulainen, R. (2019). Evolutionary neural autoML for deep learning. *Proceedings of the genetic and evolutionary computation conference* (pp. 401–409). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3321707.3321721

[109] Such, F. P., Madhavan, V., Conti, E., Lehman, J., Stanley, K. O., & Clune, J. (2017). *Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning*. https://doi.org/10.48550/arXiv.1712.06567

[110] de Campos Souza, P. V., & Sayyadzadeh, I. (2025). GWO-FNN: Fuzzy neural network optimized via grey wolf optimization. *Mathematics*, *13*(7), 1–48. https://doi.org/10.3390/math13071156

[111] Ingber, L. (1993). Simulated annealing: Practice versus theory. *Mathematical and computer modelling*, *18*(11), 29–57. https://doi.org/10.1016/0895-7177(93)90204-C

[112] Aljarah, I., Faris, H., & Mirjalili, S. (2018). Optimizing connection weights in neural networks using the whale optimization algorithm. *Soft computing*, *22*(1), 1–15. https://doi.org/10.1007/s00500-016-2442-1

[113] Stanley, K. O., Clune, J., Lehman, J., & Miikkulainen, R. (2019). Designing neural networks through neuroevolution. *Nature machine intelligence*, *1*(1), 24–35. https://doi.org/10.1038/s42256-018-0006-z

[114] Kuncheva, L. I. (2014). *Combining pattern classifiers: Methods and algorithms*. Wiley Online Library. https://doi.org/10.1002/9781118914564

[115] Brown, G. (2011). Ensemble learning. In *Encyclopedia of machine learning* (pp. 312–320). Springer. https://doi.org/10.1007/978-0-387-30164-8_252

[116] Zhou, Z. H., Wu, J., & Tang, W. (2002). Ensembling neural networks: Many could be better than all. *Artificial intelligence*, *137*(1), 239–263. https://doi.org/10.1016/S0004-3702(02)00190-X

[117] Oliveira, L. S., Sabourin, R., Bortolozzi, F., & Suen, C. Y. (2003). A methodology for feature selection using multiobjective genetic algorithms for handwritten digit string recognition. *International journal of pattern recognition and artificial intelligence*, *17*(06), 903–929. https://doi.org/10.1142/S021800140300271X

[118] LeDell, E., & Poirier, S. (2020). H2o autoML: Scalable automatic machine learning. *7th ICML workshop on automated machine learning* (pp. 1-16). International Machine Learning Society (IMLS). https://www.automl.org/wp-content/uploads/2020/07/AutoML_2020_paper_61.pdf

[119] Mirjalili, S. (2015). How effective is the grey wolf optimizer in training multi-layer perceptrons. *Applied intelligence*, *43*(1), 150–161. https://doi.org/10.1007/s10489-014-0645-7

[120] Gharehchopogh, F. S., & Gholizadeh, H. (2019). A comprehensive survey: Whale optimization algorithm and its applications. *Swarm and evolutionary computation*, *48*, 1–24. https://doi.org/10.1016/j.swevo.2019.03.004

[121] Heidari, A. A., & Pahlavani, P. (2017). An efficient modified grey wolf optimizer with Lévy flight for optimization tasks. *Applied soft computing*, *60*, 115–134. https://doi.org/10.1016/j.asoc.2017.06.044

[122] Sutton, R. S., & Barto, A. G. (1999). *Reinforcement learning: An introduction*. MIT Press. https://mitpress.mit.edu/9780262039246/reinforcement-learning/

[123] Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary computation*, *10*(2), 99–127. https://doi.org/10.1162/106365602320169811

[124] Salimans, T., Ho, J., Chen, X., Sidor, S., & Sutskever, I. (2017). *Evolution strategies as a scalable alternative to reinforcement learning*. https://doi.org/10.1162/106365602320169811

[125] Hansen, N. (2016). *The CMA evolution strategy: A tutorial*. https://doi.org/10.48550/arXiv.1604.00772

[126] Jaderberg, M., Dalibard, V., Osindero, S., Czarnecki, W. M., Donahue, J., Razavi, A., … ., & Kavukcuoglu, K. (2017). *Population based training of neural networks*. https://doi.org/10.48550/arXiv.1711.09846

[127] Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of machine learning research*, *7*, 1–30. https://www.researchgate.net/publication/220320196

[128] García, S., Fernández, A., Luengo, J., & Herrera, F. (2010). Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information sciences*, *180*(10), 2044–2064. https://doi.org/10.1016/j.ins.2009.12.010

[129] Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & de Freitas, N. (2016). Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, *104*(1), 148–175. https://doi.org/10.1109/JPROC.2015.2494218

[130] Falkner, S., Klein, A., & Hutter, F. (2018). BOHB: Robust and efficient hyperparameter optimization at scale. *Proceedings of the 35th international conference on machine learning* (pp. 1437–1446). Proceedings of Machine Learning Research (PMLR). https://proceedings.mlr.press/v80/falkner18a.html

[131] Agrawal, T., & Choudhary, P. (2022). *Metaheuristic optimization algorithms*. Morgan Kaufmann. https://www.sciencedirect.com/book/edited-volume/9780443139253/metaheuristic-optimization-algorithms

[132] Jin, Y. (2011). Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and evolutionary computation*, *1*(2), 61–70. https://doi.org/10.1016/j.swevo.2011.05.001

[133] Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2017). LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems*, *28*(10), 2222–2232. https://doi.org/10.1109/TNNLS.2016.2582924

[134] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., … ., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems* (Vol. 30, PP. 5998–6008). Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[135] Tekkali, C., & Natarajan, K. (2023). Smart fraud detection in E-transactions using synthetic minority oversampling and binary Harris Hawks optimization. *Computers, materials, & continua*, *75*(2), 3171. https://doi.org/10.32604/cmc.2023.036865

[136] Ma, L., Liu, Y., Zhang, X., Ye, Y., Yin, G., & Johnson, B. A. (2019). Deep learning in remote sensing applications: A meta-analysis and review. *ISPRS journal of photogrammetry and remote sensing*, *152*, 166–177. https://doi.org/10.1016/j.isprsjprs.2019.04.015

[137] Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A. A., Ciompi, F., Ghafoorian, M., … ., & Sánchez, C. I. (2017). A survey on deep learning in medical image analysis. *Medical image analysis*, *42*, 60–88. https://doi.org/10.1016/j.media.2017.07.005

[138] Liashchynskyi, P., & Liashchynskyi, P. (2019). *Grid search, random search, genetic algorithm: A big comparison for NAS*. https://doi.org/10.48550/arXiv.1912.06059

[139] Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, *1*(1), 67–82. https://doi.org/10.1109/4235.585893

[140] Zhang, D., Mishra, S., Brynjolfsson, E., Etchemendy, J., Ganguli, D., Grosz, B., … ., & Perrault, R. (2024). *The 2024 AI index report*. https://hai.stanford.edu/ai-index/2024-ai-index-report?hl=en-US

[141] Wistuba, M., Rawat, A., & Pedapati, T. (2019). *A survey on neural architecture search*. https://doi.org/10.48550/arXiv.1905.01392

[142] Cobo, M. J., López-Herrera, A. G., Herrera-Viedma, E., & Herrera, F. (2011). Science mapping software tools: Review, analysis, and cooperative study among tools. *Journal of the american society for information science and technology*, *62*(7), 1382–1402. https://doi.org/10.1002/asi.21525

[143] Osaba, E., Yang, X.-S., & Del Ser, J. (2020). Traveling salesman problem: A perspective review of recent research and new results with bio-inspired metaheuristics. In *Nature-inspired computation and swarm intelligence* (pp. 135–164). Academic Press. https://doi.org/10.1016/B978-0-12-819714-1.00020-8

[144] He, X., Zhao, K., & Chu, X. (2021). AutoML: A survey of the state-of-the-art. *Knowledge-based systems*, *212*, 106622. https://doi.org/10.1016/j.knosys.2020.106622

[145] Cai, H., Zhu, L., & Han, S. (2018). *Proxylessnas: Direct neural architecture search on target task and hardware*. https://doi.org/10.48550/arXiv.1812.00332

[146] Strubell, E., Ganesh, A., & McCallum, A. (2019). Energy and policy considerations for deep learning in NLP. *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 3645–3650). Association for Computational Linguistics. https://doi.org/10.18653/v1/P19-1355

[147] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., … ., & Amodei, D. (2020). Language models are few-shot learners. *Advances in neural information processing systems* (Vol. 33, pp. 1877–1901). Neural Information Processing Systems Foundation. https://dl.acm.org/doi/abs/10.5555/3495724.3495883

[148] Ying, C., Klein, A., Christiansen, E., Real, E., Murphy, K., & Hutter, F. (2019). NAS-bench-101: Towards reproducible neural architecture search. *Proceedings of the 36th international conference on machine learning* (pp. 7105–7114). Proceedings of Machine Learning Research (PMLR). https://proceedings.mlr.press/v97/ying19a.html

[149] Zela, A., Siems, J., & Hutter, F. (2020). *Nas-bench-1shot1: Benchmarking and dissecting one-shot neural architecture search*. https://doi.org/10.48550/arXiv.2001.10422

[150] Wong, C., Houlsby, N., Lu, Y., & Gesmundo, A. (2018). Transfer learning with neural autoML. *Advances in neural information processing systems* (pp. 8356–8365). Neural Information Processing Systems Foundation. https://proceedings.neurips.cc/paper_files/paper/2018/hash/bdb3c278f45e6734c35733d24299d3f4-Abstract.html

[151] Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. (2018). Deep reinforcement learning that matters. *Proceedings of the AAAI conference on artificial intelligence* (pp. 3207–3214). Association for the Advancement of Artificial Intelligence (AAAI). https://doi.org/10.1609/aaai.v32i1.11694

[152] Lindauer, M., & Hutter, F. (2020). Best practices for scientific research on neural architecture search. *Journal of machine learning research*, *21*(243), 1–18. http://jmlr.org/papers/v21/20-056.html

[153] He, J., & Yao, X. (2001). Drift analysis and average time complexity of evolutionary algorithms. *Artificial intelligence*, *127*(1), 57–85. https://doi.org/10.1016/S0004-3702(01)00058-3

[154] Dong, X., & Yang, Y. (2020). *Nas-bench-201: Extending the scope of reproducible neural architecture search.* https://doi.org/10.48550/arXiv.2001.00326

[155] Siems, J., Zimmer, L., Zela, A., Lukasik, J., Keuber, M., & Hutter, F. (2021). NAS-bench-301 and the case for surrogate benchmarks for neural architecture search. *International conference on learning representations* (PP. 1-11). OpenReview. https://ml.informatik.uni-freiburg.de/wp-content/uploads/papers/20-NIPS_WML-NB301.pdf

[156] Mellor, J., Turner, J., Storkey, A., & Crowley, E. J. (2021). Neural architecture search without training. *Proceedings of the 38th international conference on machine learning* (pp. 7588–7598). Proceedings of Machine Learning Research (PMLR). https://proceedings.mlr.press/v139/mellor21a.html

[157] Chen, W., Gong, X., & Wang, Z. (2021). *Neural architecture search on imagenet in four gpu hours: A theoretically inspired perspective*. https://doi.org/10.48550/arXiv.2102.11535

[158] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M. A., Lacroix, T., … ., & Lample, G. (2023). *LLaMA: Open and efficient foundation language models*. https://doi.org/10.48550/arXiv.2302.13971

[159] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., ... & Chen, W. (2022). Lora: Low-rank adaptation of large language models. *International conference on learning representations (Iclr)* (Vol. 1, No. 2, p. 3). https://arxiv.org/pdf/2106.09685v1/1000

[160] McMahan, B., Moore, E., Ramage, D., Hampson, S., & y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. *Proceedings of the 20th international conference on artificial intelligence and statistics (AISTATS 2017)* (pp. 1273–1282). Proceedings of Machine Learning Research (PMLR). https://proceedings.mlr.press/v54/mcmahan17a.html

[161] Schwartz, R., Dodge, J., Smith, N. A., & Etzioni, O. (2020). Green AI. *Communication of the ACM*, *63*(12), 54–63. https://doi.org/10.1145/3381831

[162] Zhang, G. (2011). Quantum-inspired evolutionary algorithms: A survey and empirical study. *Journal of heuristics*, *17*(3), 303–351. https://doi.org/10.1007/s10732-010-9136-0

[163] Aleti, A., & Moser, I. (2016). A systematic literature review of adaptive parameter control methods for evolutionary algorithms. *ACM computing surveys*, *49*(3), 1–35. https://doi.org/10.1145/2996355

[164] Li, K., Fialho, Á., Kwong, S., & Zhang, Q. (2014). Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition. *IEEE transactions on evolutionary computation*, *18*(1), 114–130. https://doi.org/10.1109/TEVC.2013.2239648

[165] Gaier, A., & Ha, D. (2019). Weight agnostic neural networks. *Advances in neural information processing systems* (Vol. 32, PP. 5365–5379). Curran Associates, Inc. https://proceedings.neurips.cc/paper/2019/hash/e98741479a7b998f88b8f8c9f0b6b6f1-Abstract.html

[166] Thornton, C., Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2013). Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. *Proceedings of the 19th ACM sigkdd international conference on knowledge discovery and data mining* (pp. 847–855). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/2487575.2487629

[167] Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., & Hutter, F. (2015). Efficient and robust automated machine learning. *Advances in neural information processing systems* (Vol. 28, PP. 2962–2970). Curran Associates, Inc. https://proceedings.neurips.cc/paper/2015/hash/11d0e6287202fced83f79975ec59a3a6-Abstract.html