Testing the Effectiveness, Efficiency, and Stability of Search-Based Hyperparameter Optimizers

Fernando Berti Cruz Nogueira, Kelvin Mock

Abstract: Hyperparameter optimization (HPO) is a critical but computationally expensive task in the development of AI models. While numerous search-based algorithms exist to automate this process, a holistic analysis of the trade-offs between solution quality, computational efficiency, and search stability is less common. We formulate our project to apply and compare metaheuristic search algorithms against a randomized search baseline for HPO. Our analysis will assess three quality metrics: the performance of the resulting model, the computational cost required to find a solution, and the consistency of results across runs. We will test these algorithms using the grayscale CIFAR-10 dataset on a selection of model architectures with reference to some papers (Bibaeva, 2018) (Catalin Stoean, Miodrag Zivkovic, AleksandraNebojsa Bacanin, Roma Strulak-Wójcikiewicz, Milos Antonijevic, and Ruxandra Stoean, 2023).

Introduction

The Overall Idea of the Project

Artificial Intelligence now spans its applications across multiple domains, relying heavily on machine learning models. A typical model is a complex mathematical formula, such as a polynomial, that consists of weights that act as coefficients for each term. The process of training involves finding the most suitable vector of these weights (known as parameters) for a given formula and dataset. However, this training process itself is controlled by another set of variables called hyperparameters.

Hyperparameters dictate how a model learns and include critical settings like the learning rate, batch size per iteration, and the number of epochs. Tuning these hyperparameters is a demanding task, as it is very difficult to determine the optimal combination for a given model. Finding this optimal set through manual trial-and-error is often impractical, as it would require running the full resource-intensive training process for every possible combination. This challenge of finding the best hyperparameters is a significant bottleneck in developing effective AI models.

Our project will conduct a study to test and compare the quality attributes of different search-based optimizers: the quality of the solution found, the computational cost to find it, and the consistency of results across runs. The experiments will be conducted using grayscale images and a selection of model architectures.

Why the Project is Significant

As complex neural networks are now widely applied in critical domains, from energy generation and power prediction (Catalin Stoean, Miodrag Zivkovic, AleksandraNebojsa Bacanin, Roma Strulak-Wójcikiewicz, Milos Antonijevic, and Ruxandra Stoean, 2023) to bioinformatics and clinical diagnosis using medical imaging (Sajjad Nematzadeh, Farzad Kiani, Mahsa Torkamanian-Afshar, Nizamettin Aydin, 2021), the need for a reliable model development process has increased. Hyperparameter optimization is a key step in this process that directly influences a model's final performance.

While metaheuristics have shown promise for improving HPO, the choice of a specific algorithm still presents a difficult trade-off. Evaluations, such as the comprehensive study by Sajjad et al. (Sajjad Nematzadeh, Farzad Kiani, Mahsa Torkamanian-Afshar, Nizamettin Aydin, 2021), often compare algorithms based on effectiveness (i.e., the quality of the final solution), and efficiency (i.e., the computational cost to find it). However, the stability of the results is less frequently analyzed. An optimizer may be effective and efficient on average but produces highly inconsistent results across different runs. Knowing that an optimizer is reliable and will perform predictably is essential to its effectiveness.

An assessment of the trade-offs between effectiveness, efficiency and stability is needed for practitioners to select the appropriate optimizer for their specific needs, yet this multi-objective analysis is less common. Our project addresses this by conducting a study to compare search-based optimizers across these three quality attributes. The goal is to provide a comparative analysis of their performance profiles, helping to answer a more practical engineering question about their expected cost and reliability.

Relevancy: how the project is relevant to SBSE or to the applications SMT/SAT solvers for ST or SE

The process of tuning hyperparameters is a significant software engineering challenge where SBSE offers a structured approach. Common baseline methods for this task, like grid and random search, have well-known limitations. An exhaustive grid search, for one thing, is often computationally unfeasible for more complex models, and at the same time, reliance on a discrete grid can easily overlook optimal values, especially on continuous variables. While random search is more tractable, its unguided nature means the quality of a solution depends heavily on chance. These trade-offs motivate the need for more advanced search-based algorithms and a framework to properly evaluate them.
As an effective alternative, many studies support the use of metaheuristic algorithms. The landscape of these algorithms is vast as shown by the mind map below.

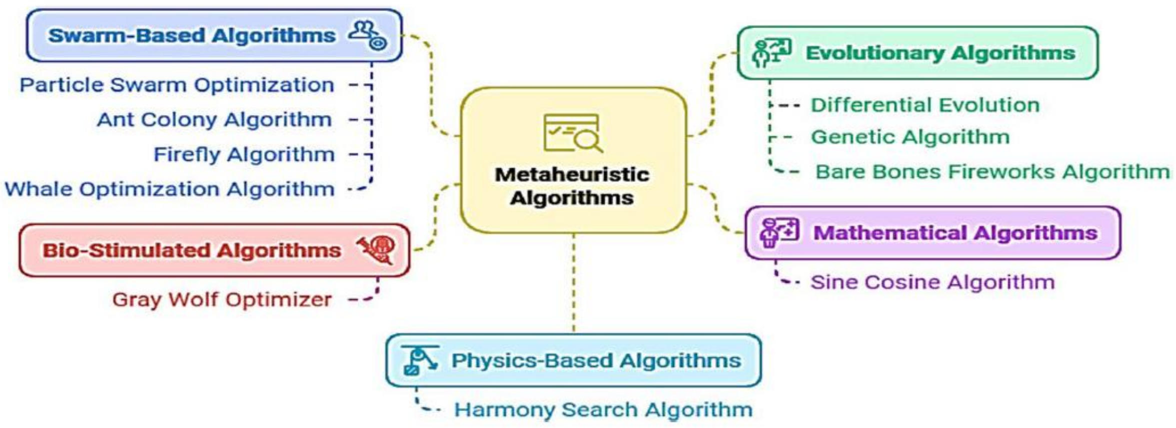

Figure 1 Metaheuristics Overview (Mohammed Q Ibrahim, Nazar K. Hussein, David Guinovart, Mohammed Qaraad, 2025)

These algorithms employ sophisticated heuristics to navigate complex search spaces more intelligently than simple baseline methods (Mohammed Q Ibrahim, Nazar K. Hussein, David Guinovart, Mohammed Qaraad, 2025). This variety, however, introduces a problem: with so many options, how do we select the most appropriate algorithm for a given task? Our work will conduct a comparative study that assesses these optimizers as tools, focusing on a balanced view of their quality attributes: effectiveness of the solutions they find, their computational efficiency, and the stability of their performance across many runs.

We will focus on two of the most prominent families of these metaheuristics. The first are the Evolutionary Algorithms, such as the Genetic Algorithm (GA), which simulate principles of natural selection through operators like selection, crossover, and mutation to evolve a population of candidate solutions. The second are Swarm Intelligence Algorithms, such as Particle Swarm Optimization (PSO), which models the collective emergent behaviour of decentralized systems (such as a flock of birds) to converge on an optimal solution.

Related Work

Our project builds on Bibaeva’s work (Bibaeva, 2018), which introduced an evolutionary memetic algorithm for hyperparameter optimization, and compared it to simulated annealing, focusing primarily on solution quality. We extend this work by evaluating not only effectiveness but also efficiency and stability across multiple runs. Additionally, we draw from Sajjad et al. (Sajjad Nematzadeh, Farzad Kiani, Mahsa Torkamanian-Afshar, Nizamettin Aydin, 2021), who explored metaheuristics in biomedical applications, highlighting the need for reliable and consistent optimization methods. Our proposal aims to provide a more holistic comparison of search-based optimizers by incorporating these additional quality metrics.

Research Questions

1. How do representative metaheuristic algorithms compare against a randomized search baseline in terms of effectiveness and efficiency when performing HPO prior to training?
2. What is the difference in performance stability between the selected metaheuristic algorithms and traditional solutions like the randomized search baseline?

Data Set and Replication plan

To test the effectiveness, efficiency and stability of different optimizers, this project will use the CIFAR-10 dataset. This dataset is chosen because it is a widely recognized benchmark in machine learning, providing a non-trivial challenge for model optimization and allowing for a straightforward comparison of our results with existing studies, such as the work by Bibaeva (Bibaeva, 2018).

The CIFAR-10 dataset (Krizhevsky, 2009) consists of a collection of 60,000 32×32 color images distributed across 10 object classes (e.g., airplane, cat, truck). For the purposes of our experiments, all images will be converted to a grayscale format ($32 \times 32 \times 1$) to manage computational complexity.

Overview

Our hyperparameter optimization problem is formulated with a representation, fitness function, and operators. A state, or a candidate solution in the search space, is a specific configuration of hyperparameters for a given model architecture. Each candidate solution is evaluated by training a model with its corresponding hyperparameters on the CIFAR-10 dataset and measuring the resulting performance.

To reduce noise, it is preprocessed under specific procedures before training and evaluating the model. The following preprocessing steps are inspired by Kelvin's previous work (Mock, Texture-Image-Comparison, 2024) at his previously taken course – CSI5341 Learning-based Computer Vision at the University of Ottawa:

1. The image is read into a digital array of pixel values.
2. The image is converted to grayscale (in the shape of $32 \times 32 \times 1$).
3. The pixel values are normalized to a range of $[0,\ 1]$ by dividing by 255.

A multi-objective fitness function will then be used to formulate the quality of a certain set of hyperparameters on a particular model and a particular dataset. Traditionally, we measure the accuracy, precision, recall, and (macro and micro) F1 scores of a certain array of predictions from a model. Those metrics will be a significant part of the fitness function.

Operators depend on algorithms. Evolutionary algorithms like the genetic algorithm or memetic algorithm uses *selection*, *crossover*, and *mutation*. Simulated annealing uses an exponential cooling schedule to adjust the temperature, which is an important parameter to evaluate a solution with a fitness function. Other algorithms whose performance is worth investigating include those from swarm optimization, namely the particle-based and ant colony algorithms. Particle Swarm Optimization (PSO) takes a *velocity update* to adjust each particle's velocity using inertia, cognitive, and social coefficients; and position update to move each particle according to its updated velocity (Mohammed Q Ibrahim, Nazar K. Hussein, David Guinovart, Mohammed Qaraad, 2025). Ant Colony Optimization uses mainly Pheromone deposition and Pheromone evaporation.

The representation of a candidate solution will be a numerical vector tailored for each model architecture.

To answer our research questions, we execute the following steps for each optimizer:

- We will perform N independent runs to account for the stochastic nature of the algorithms.
- To measure effectiveness, we will analyze the distribution (mean, median, and best) of the final fitness scores achieved across the N runs.
- To measure efficiency, we will use two complementary metrics to ensure a fair comparison:
    - We will plot the best-found fitness score as a function of the number of fitness evaluations used to show how quickly an algorithm learns.
    - We will record the total wall-clock execution time for each run to capture the real-world cost, including the algorithm overhead.
- To measure stability, we will calculate the variance of the final fitness scores across the N runs.

Analysis or implementation Procedure

Our procedure will begin by implementing the necessary models and search algorithms. Based on model categorizations in prior work (Mock, Drug-Consumption-Machine-Learning-analysis, 2024), we will use three architectures: a Decision Tree, *k*-Nearest-Neighbours, and a Convolutional Neural Network. We will also implement two metaheuristic optimizers and a Randomized Search baseline. The experimental process will be automated to manage the independent runs and log the required performance data.

The analysis will use this data to answer our research questions. Effectiveness will be assessed by analyzing the distribution of final fitness scores with summary statistics and box plots. Efficiency will be evaluated using convergence plots that track the best fitness found against both the number of evaluations and wall-clock time. Stability will be quantified by calculating the variance of the final fitness scores across the independent runs.

Teamwork Plan

	Fernando	**Kelvin**
Phase 1: Project Setup	Extract images from the CIFAR-10 dataset, and perform necessary preprocessing: grayscale conversion, normalization.	Implement machine learning models and a multi-objective fitness function for evaluation.
	Implement shared interfaces for the optimizers.	
Phase 2: Experiments	Implement and evaluate a representative Swarm Intelligence algorithm.	Implement and evaluate the randomized search baseline.
		Implement and evaluate a representative Evolutionary algorithm.

References

Bibaeva, V. (2018). USING METAHEURISTICS FOR HYPER-PARAMETER OPTIMIZATION OF CONVOLUTIONAL NEURAL NETWORKS. *2018 IEEE INTERNATIONAL WORKSHOP ON MACHINE LEARNING FOR SIGNAL PROCESSING, SEPT. 17–20, 2018, AALBORG, DENMARK.* Hamburg.

Catalin Stoean, Miodrag Zivkovic, AleksandraNebojsa Bacanin, Roma Strulak-Wójcikiewicz, Milos Antonijevic, and Ruxandra Stoean. (2023, March 4). Metaheuristic-Based Hyperparameter Tuning for Recurrent Deep Learning: Application to the Prediction of Solar Energy Generation. *Advances in Mathematics for Applied Machine Learning*.

Krizhevsky, A. (2009). *The CIFAR-10 dataset*. Retrieved from Alex Krizhevsky: https://www.cs.toronto.edu/~kriz/cifar.html

Mock, K. (2024, December 9). *Drug-Consumption-Machine-Learning-analysis*. Retrieved from GitHub: https://github.com/kmock930/Drug-Consumption-Machine-Learning-analysis

Mock, K. (2024, October 23). *Texture-Image-Comparison*. Retrieved from GitHub: https://github.com/kmock930/Texture-Image-Comparison

Mohammed Q Ibrahim, Nazar K. Hussein, David Guinovart, Mohammed Qaraad. (2025). Optimizing Convolutional Neural Networks: A Comprehensive Review of Hyperparameter Tuning Through Metaheuristic Algorithms. *Computational Methods in Engineering*.

Sajjad Nematzadeh, Farzad Kiani, Mahsa Torkamanian-Afshar, Nizamettin Aydin. (2021). Tuning hyperparameters of machine learning algorithms and deep neural networks using metaheuristics: A bioinformatics study on biomedical and biological cases. *Computational Biology and Chemistry*.