Analyzing and Rating Greenness of Nature-Inspired Algorithms

Chander Jindal [1*†], Kanaishk Garg [1†], Shobhit Kumar [1†]

[1] Department of Computer Science and Engineering, Maharaja Agrasen Institute of Technology, Delhi, India

*** Correspondence:**

Chander Jindal

chanderjindal2@gmail.com

†These authors contributed equally to this work and share first authorship

Abstract: One could say that machine learning is the secret to using data analysis to inform decisions. With thus much utilization, it becomes crucial that these algorithms use the least number of resources possible in order to save down on ongoing expenses and deliver effective results faster. An optimizer is a technique or procedure to modify the various parameters that can more efficiently reduce the loss. The Nature-Inspired Optimization (NIO) algorithm is one such optimizer. They are very effective at solving multi-dimensional and multi-modal issues with optimal solutions. Due to their significance and importance, numerous nature-inspired optimization algorithms (NIOAs) have been presented during the past few decades.

This paper offers a critical evaluation of energy use and the associated carbon footprint for a few well-known NIO algorithms. Microsoft Joulemeter is used to measure the energy used while each algorithm is running. The associated carbon footprint of each method, however, is determined using the recommendations of India's Central Electricity Authority. According to the study's findings, each algorithm uses energy in a different way to accomplish the same task. The results of this study will aid software designers in selecting better (greener) NIO algorithms from the available possibilities. In order to identify the most environmentally friendly NIO algorithms, future study might consider more NIO algorithms and their modifications for energy consumption analysis. Additionally, further investigation into the potential effects of different CPU architectures on the efficiency and power usage of the NIO algorithms may be taken into account.

Keywords: microsoft joulemeter; central electricity authority of india; nature-inspired optimization techniques; energy consumption; carbon footprint; green software; environmental effect

1. Introduction

Changing an existing process to enhance the likelihood of favorable results and decrease the likelihood of undesirable ones is the broad definition of optimization. In many different disciplines, including engineering, corporate operations, industrial designs, etc., optimization is a common mathematical problem. Optimizations could be of different types such as lowering energy costs or raising performance and efficiency. The majority of traditional optimization algorithms used to address real-world problems are highly non-linear, have several local optima, and involve complex nonlinear constraints [1]. Contrarily, NIO algorithms are population-based metaheuristics that replicate a wide range of natural phenomena [2]. In contrast to conventional optimization techniques, they are successful in avoiding local optima. As a result, they are extensively employed in a variety of sectors,

including manufacturing, environmental engineering, finance, biology, data mining jobs, etc., to handle highly nonlinear optimization problems.

Because of the popularity of mobile and IoT devices, computing energy utilization should be taken into account while developing programs for high performance and mobile software applications. Software programs may become more eco- and energy-friendly as a result of enhanced algorithms and data structures. Runtime, which served as the benchmark for measuring an algorithm's performance in earlier decades, was the only performance indication considered for study and optimization of an algorithm [3]. Energy consumption has increased as a result of the recent rapid development of high-performance computers and embedded devices with faster processors. Therefore, it is crucial to consider an algorithm's energy usage when evaluating it (i.e., in terms of performance and sustainability). An algorithm's effectiveness and efficiency must be assessed in the context of a particular application since how it is implemented will have an influence on energy consumption and the environment.

One approach for assessing the ecological consequences of computers and other computing devices is through carbon footprint [4] by evaluating a program's degree of power efficiency related to carbon footprint and implementing it into ecologically friendly company operations or procedures organizations may make the application a crucial component of their corporate social responsibility activities. Machine learning model deployment has grown massively in recent years [5]. Considerable issues have emerged about the energy usage and expense related to developing ML models and training them [6]. Therefore, it's indeed crucial to consider an application's carbon footprint while planning, constructing, as well as deploying it.

More than a hundred NIO algorithms and their variations are now known and available in the literature [2]. The Bat Algorithm (BAT), Camel Algorithm (CAM), Cuckoo Search (CS), Firefly Algorithm (FIR), and Particle Swarm Optimization (PAR) are some of the regularly utilized NIO algorithms that will be the subject of this study's examination of energy consumption and associated carbon footprint. These algorithms were taken into account for this study due to the wide range of applications for them. This work aims to demonstrate how one may experimentally assess the energy consumption of different algorithms. Keep in mind that future research might concentrate on different NIO algorithms.

Bat Algorithm (BAT/BA): There are around 1000 species of bats. The Bat Algorithm (BA) is based on the Echolocation behavior of microbats [7]. Microbats are medium-sized bats that eat insects. They used a SONAR technique called echolocation to detect prey. Artificial bats that imitate actual bats' natural pulse loudness and emission rate serve as search agents in the search process carried out by the Bat Algorithm. Furthermore, it aids in undertaking global optimization since it uses a meta-heuristic approach [8]. In a variety of fields, including data mining, big data, and machine learning, BA has been used to address challenging issues.

Camel Algorithm (CAM/CA): a cutting-edge optimization method motivated by camel migration patterns in the desert and other challenging situations. A camel will often travel toward an area with food and water. In light of that, a number of variables and operators are taken into account to outline the CA algorithm procedure, including the temperature effect, the supply (water and food), the camel endurance, the camel visibility (and / or hearing) range, random walk, the group effect (multi-solution), the termination condition (dying or moving back), the land conditions (oasis, quick sand, storms, etc.), and limitations (max speed, age and carrying weight). The camel algorithm simple structure along with its efficient search ability allow it to deal effectively with unimodal and multimodal test functions to find an optimal solution even with difficult ones [9].

Cuckoo Search (CUC/CS): Xin-She Yang and Suash Deb created the optimization algorithm cuckoo search in 2009 [10]. The obligate brood parasitism of some cuckoo species, which involves the cuckoos' laying their eggs in the nests of host birds of other species, served as its model. Direct combat between some host birds and the trespassing cuckoos is possible. For instance, if the host bird learns the eggs are not its own, it may either discard the alien eggs or quit the nest and make a new one elsewhere. Some cuckoo species, like the New World brood-parasitic Tapera, have developed in such a way that female parasitic cuckoos are frequently extremely skilled at mimicking the colours and patterns of the eggs of a select few host species. Such breeding behaviour was idealised by cuckoo search, which can be used to solve numerous optimization issues [11].

Firefly Algorithm (FIR/FA): The glowing pattern that firefly swarms exhibit served as inspiration for FA [2]. FA is incredibly flexible and easy to use. It is based on the ideas that the attractiveness and the brightness are inversely correlated and that Fireflies are attracted to one another, if two fireflies have the same brightness. The software creates creative approaches and continues to search solution space. The Random Walk unpredictability factor refers to this. There are several applications for FA, including image compression, antenna design optimization, classification, feature selection, etc.

Particle Swarm Optimization (PAR/PSO): The concept of Particle Swarm Optimization (PSO) is inspired by the swarm intelligence of fish schooling and bird flocks that are in search of food [12]. Each particle in these groups has its own velocity and position. It has been applied to solve a variety of issues, including data clustering [17], human motion tracking [14], cloud resource allocation [15], assembly line balance [16], and cost prediction for software [13]. However, a drawback of PSO is that it has a poor rate of convergence during the iterative process and is prone to falling into local optimum in high-dimensional space [18].

The objectives of our study are as follows.

1. Conduct a critical review of the literature on the effects of the Information and Communications Technology (ICT) on the environment, green or energy-efficient programs, the effects of software's power usage on hardware, the analysis of software's consumption of electricity in algorithm implementations, and energy-efficient and nature-inspired algorithms

2. Implement the above-stated NIO Algorithms using Python programming language and the class NatureInspiredSearchCV provided by the sklearn_nature_inspired_algorithms library and NiaPy for nature inspired algorithms, a series of experiments to determine how much energy each method consumes is carried out by Bayesian Optimization.

3. Based on how much energy each technique uses, calculate its equivalent carbon footprint.

Numerous previous works have compared the energy usage of various programming languages [19–21] and sorting algorithm implementations [22–25]. Only one other team has evaluated the NIO algorithms' energy usage and greenness. The effectiveness of Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Differential Evolution (DE), and Artificial Bee Colony (ABC) was assessed [26]. According to their results Differential Evolution is the most efficient but that result cannot be directly compared to this study due to use of different methodology to test the algorithm suite. Still, we hope that this study will assist programmers in selecting the greenest nature inspired algorithms to address a certain domain problem where minimizing energy usage is of the greatest priority.

The rest of this article is divided into the following sections: Section 2 presents a review of the literature on the effects of ICT on the environment, green or energy-efficient software, the influence of hardware energy consumption on software, and related works on the analysis of energy consumption in algorithm implementations as well as on nature-inspired algorithms and

energy efficiency. The brief introduction of techniques in Section 3 covers both macro and micro methodology as well as the planning and design of experiments. Section 4 presents the findings and commentary, which goes through the ethical concerns and difficulties of this study as well as the energy usage and associated carbon footprint of each algorithm. Section 5 provides a summary of the topic and suggestions for additional research.

2. Literature Review

ICT sector is responsible for between 2.1 and 3.9% of the world's carbon emissions, with the remaining 97.9 to 96.1% coming from other industries including transportation and agriculture [27]. The environment and the economy would suffer as a result of the rise in carbon emissions brought on by Greenhouse Gases and other causes [28]. The ICT sector can significantly contribute to lowering global carbon emissions by reducing the carbon footprints of its products and services because there is growing global demand for ICT goods and services. Energy-efficient hardware and other embedded systems have been the subject of extensive study [29–32], but software and application development should also receive significant attention [33–34].

a. Energy-efficient software

Green or energy-efficient software is defined as using less energy for effective computing while causing little environmental harm [35]. The energy efficiency of web-based software applications and software features has been the subject of numerous research [36, 37]. Software can easily be estimated to use between 25% and 40% of the total energy used by a device, depending on the laptop or mobile battery [38]. However, because it is correlated with the host device's life cycle, the indirect impact of software is more challenging to measure [39]. Only when both the positive and negative impacts are adequately taken into consideration throughout the design and deployment phases can energy-efficiency of a software be truly accomplished. In light of this, optimising ICT application services is essential to lowering harmful environmental effects.

b. Software’s Impact on Hardware-Related energy consumption

How much energy hardware uses and how long a device's battery lasts are strongly impacted by software's energy usage patterns [40]. A device's energy usage may eventually increase if a software or application that is poorly built disables various hardware-based energy-saving capabilities [41]. For instance, it can prevent hardware from using energy-saving features and impact how the hardware is used, which could ultimately result in an increase in indirect energy usage [42]. One of the trickiest tasks during the design stage of an embedded system is the development of energy-efficient software that enhances the energy efficiency of a piece of hardware. Various trade-offs between productivity and sustainability will need to be considered in order to increase software and application productivity while maintaining energy efficiency [43].

c. Analysis of Energy Consumption in algorithms implementations

Rashid and colleagues [22] evaluated the energy efficiency of four sorting algorithms, including Bubble, Merge, Quick, and Counting sort. On an ARM-based device, an experiment was put up to measure the energy consumption of four sorting algorithms created in three different programming languages. This study found that the ARM assembly language version of the Counting sort was the most ecologically efficient option.

Energy usage for five sorting algorithms—Bubble, Insertion, Quick, Selection, and Counting sort—has been calculated in [23]. Five distinct Apps, one for each sorting method, were made to track energy usage in this investigation. This study found that Quick sort is the most energy-efficient sorting method in common scenarios, while Bubble sort is the most energy-intensive algorithm.

Using C language implementation, Deepthi and colleagues conducted tests to examine the effects of various sorting algorithms on energy consumption [24]. This study discovered that the effectiveness of these sorting algorithms is influenced by both time and energy. Six sorting algorithms were taken into consideration in this study: Quick, Merge, Shell, Insertion, Selection, and Bubble sort. Quick, Merge, and Shell sort have been found to use similar amounts of energy, however Insertion and Selection sorts use far less energy than Bubble sort.

Ayodele and colleagues performed a comparative experimental analysis of the energy consumption of these three algorithms using three programming languages (C, Java, and Python), two algorithm implementation styles (Iterative and Recursive), and three algorithm types (Quick, Merge, and Insertion) [25]. The size of the data, the programming language used, and the way the algorithms are implemented, according to this study, all affect how much energy is utilized. This study also provides recommendations for picking the sorting algorithm type and its implementation style in order to reduce energy consumption.

In research conducted by Jamil and Kor, energy consumption of a few nature-inspired algorithms has been analyzed on a dataset [26]. Particle swarm optimization, differential evolution, the artificial bee colony algorithm, and the genetic algorithm were all employed. Differential Evolution (DE), which is shown to be the most environmentally friendly among other optimization algorithms, uses much less energy than each of the other methods.

d. Nature-inspired algorithms and energy efficiency

Existing nature-inspired algorithms research primarily addresses the following areas of research: optimization [1, 2, 44, 45] using metaheuristics [46] or heuristics approaches [47]; greening processes, for example greening the supply chain [48], smart energy management [49], data center energy efficiency [50]; energy efficiency [51] and energy optimization [52] in wireless sensor network clustering. Since there are currently few studies on energy-efficient, nature-inspired algorithms, as our critical literature evaluation has indicated, our research intends to address this issue in order to further encourage research in this area.

3. Research / Methodology

The next section will go through the various tools and software that are used in the study.

- **Macro Methodology**

The initial step includes data collection and its subsequent preprocessing. Afterward, the accuracy, energy consumption, and carbon footprint of the five algorithms, that are BAT, CUC, FIR, CAM, and PAR(PSA) are estimated and analyzed. The findings are used to draw conclusions that specify the algorithm having the best ratio of Accuracy to Energy Consumption. Lastly, all the results are summarized and discussed.

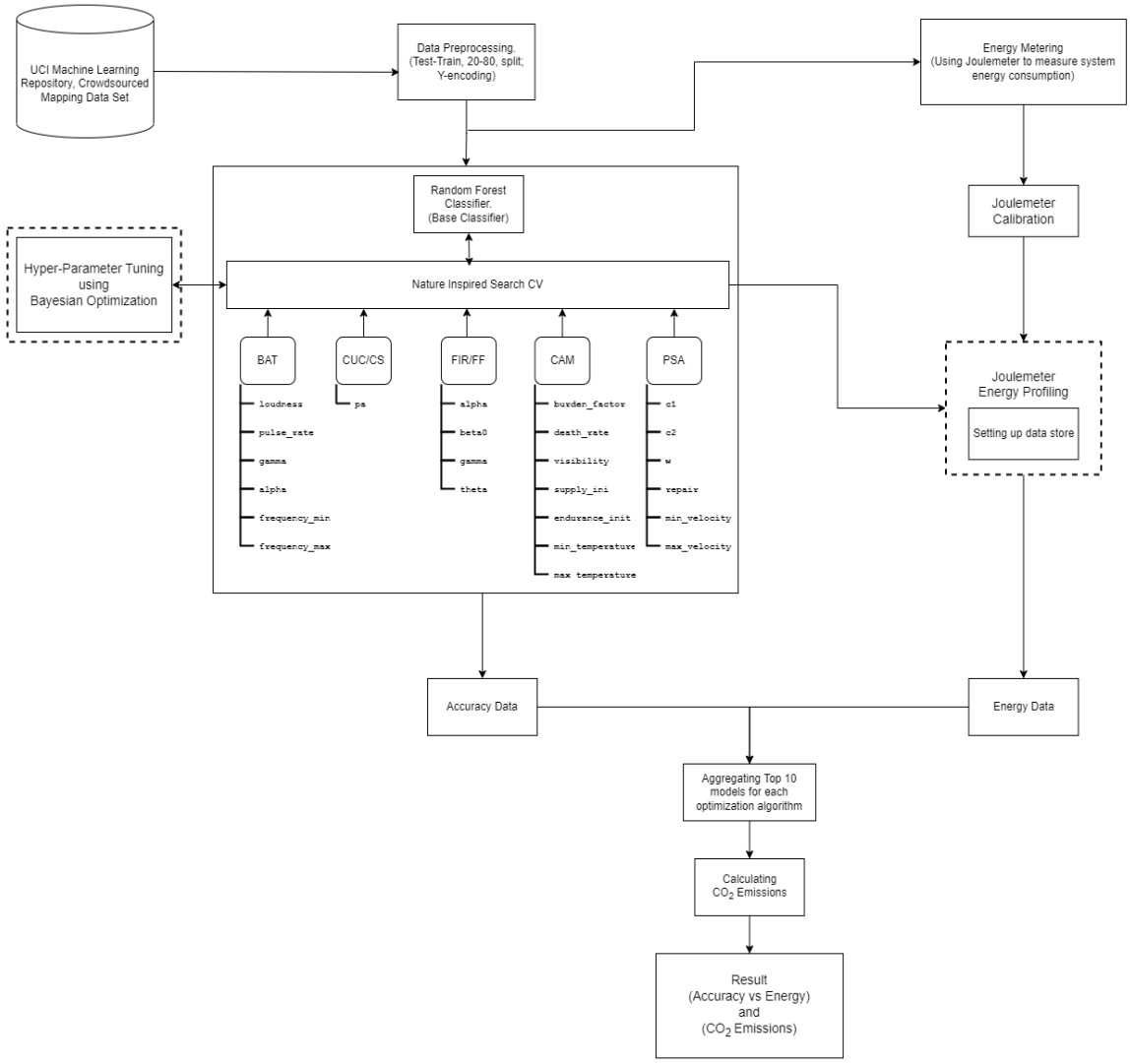

Figure 1. Methodology

- **Micro Methodology**

a. Data Availability:

The dataset used in this study has been obtained from the UCI machine learning repository. It's called the Crowdsourced Mapping Data Set. Automated classification of satellite pictures into several land cover classifications (impervious, farm, forest, grass, orchard, and water) is done using crowdsourced data from OpenStreetMap [53].

The NIO methods taken into consideration in this study were implemented accordingly using Python programming language with the sklearn-nature-inspired-algorithms, a machine learning library [54] and NiaPy, a library dedicated to Nature Inspired Optimization Algorithm in Python [55].

b. Energy Profiling:

Microsoft Joulemeter software [56], which can monitor the energy used by a running application or software as well as by specific hardware resources, including CPU, Monitor,

Disk, and Idle or Base power, was used to compute the expected energy consumption of each NIO technique.

c. Carbon Footprint:

For the purpose of calculating carbon emissions, the Central Electricity Authority of India's recommendations have been followed [57]. After collecting the amount of energy used for an experiment (in kWh), the data is converted to the equal amount of carbon released using the formula below.

CO2 Emissions = 0.85* E(kW-hr/year) where E is the energy consumed.

1kWhr of Energy Consumed = 0.85Kg of CO_2 emission

72 Joules = 17 mg of CO_2 emissions

- Experiment Setup

a. System Specification:

Different hardware requirements would produce various outcomes. Consequently, a laptop with the following characteristics was used for all experiments:

Specification of Laptop Used	
Model	Lenovo Ideapad 530S
Operating System	Windows 10 (19043.2006)
Processor	Intel® Core™ i5-8250U CPU @ 1.60Hz
RAM	8 GB
Storage	256 GB

Table 1. System Specification

b. Calibrating Joulemeter

In case autocalibration (in Joulemeter) does not work one will be needed to manually calibrate Joulemeter to get the required power readings.

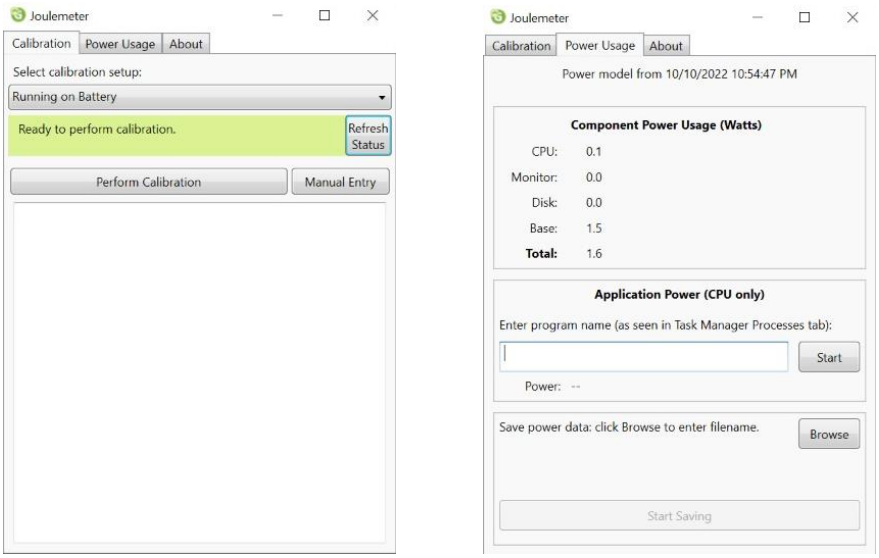

Figure 2. Calibration of Joulemeter

c. Experiment design:

First calibrated Microsoft Joulemeter on the specified system. Next, we implemented a random forest classifier as our base classifier and used the NIO algorithms to optimize our results. We then calculated CPU energy consumption for each of the instances for all the different algorithms by changing the number of decision trees. Joulemeter can evaluate the power usage up to 0.1 watts. As such, any error by the software is 0.1 watts. The results obtained were plotted in the form of graphs using Pandas.

Each algorithm's related result has been kept in its own CSV file. For the purpose of comparing the energy usage of these five methods, all of the results from each algorithm have been combined in an Excel file. The experiment design can be summed up as follows.

- Algorithms for nature-inspired optimization (NIO): PAR, CUC, CAM, BAT, FIR
- Programming Language: Python
- Benchmark Function: Sphere Function
- Space for Search: [-5.12, 5.12]

4. Findings:

The details of energy consumed by each optimization algorithm for top 10 parameter sets are shown in Table 2, while Table 3 depicts the average power, accuracy and CO_2 emission of given models.

Name of Algorithm	Keys	Time Taken(s)	Total Energy(J)	CPU Energy(J)	Disk Energy(J)	Base Energy(J)	Power Consumption(W)
Bat Algorithm	BAT-0042	100	323.20	177.20	0.0	150.0	3.232
	BAT-0034	793	2728.30	1561.6	0.1	1189.5	3.440
	BAT-0008	76	247.1	136.40	0.0	114.0	3.251
	BAT-0029	230	761.90	427.20	0.0	345.0	3.313
	BAT-0011	78	251.80	138.5	0.0	117.0	3.228
	BAT-0017	533	1741.1	964.30	0.0	799.5	3.267

	BAT-0003	306	980.20	535.40	0.0	459.0	3.203
	BAT-0040	150	490.00	270.80	0.0	225.0	3.267
	BAT-0016	115	371.60	204	0.0	172.5	3.231
	BAT-0020	9	29.4	16.4	0.0	13.5	3.267
Camel Algorithm	CAM-0000	164	527.00	287.30	0.3	264.0	3.213
	CAM-0049	704	2264.00	1241.10	0.0	1056.0	3.215
	CAM-0024	222	708.40	384.10	0.0	333.0	3.191
	CAM-0003	164	529.00	289.40	0.0	246.0	3.226
	CAM-0026	97	313.80	172.60	0.0	145.5	3.235
	CAM-0008	19	61.3	33.5	0.0	28.5	3.226
	CAM-0041	38	123.7	69.10	0.0	57.0	3.255
	CAM-0033	840	2695.80	1475	0.6	1260.0	3.209
	CAM-0025	124	398.60	218.50	0.0	186.0	3.214
	CAM-0006	53	172.80	95.70	0.0	79.5	3.260
Cuckoo Search	CUC-0009	16	51.10	27.70	0.0	24.0	3.194
	CUC-0031	264	852.80	468.30	0.0	396.0	3.230
	CUC-0010	84	269.3	146.50	0.0	126.0	3.206
	CUC-0042	678	2185.10	1198.30	0.0	1017.0	3.223
	CUC-0035	40	130.0	71.90	0.0	60.0	3.25
	CUC-0040	612	1978.50	1086.70	0.1	918.0	3.233
	CUC-0041	258	832.00	455.3	0.0	387.0	3.225
	CUC-0039	63	206.30	114.50	0.0	94.5	3.275
	CUC-0046	84	273.6	151.30	0.0	126.0	3.257
	CUC-0048	69	222.40	122.40	0.0	103.5	3.223
Firefly Algorithm	FIR-0042	340	1045.90	551.20	0.0	510.0	3.076
	FIR-0037	218	709.10	388.80	0.0	327.0	3.253
	FIR-0003	933	3013.40	1654.10	0.0	1399.5	3.230
	FIR-0049	93	288.2	152.8	0.0	139.5	3.099
	FIR-0043	527	1623.00	855.80	0.0	790.5	3.080
	FIR-0041	129	397.70	209.50	0.0	193.5	3.083
	FIR-0014	74	238.5	131.50	0.0	111.0	3.223
	FIR-0009	119	386.5	212.30	0.0	178.5	3.248
	FIR-0012	86	280.20	154.70	0.0	129.0	3.258
	FIR-0000	92	295.70	162.2	0.0	138.0	3.214
Particle Swarm Optimization	PAR-0016	186	609.60	338.9	0.0	279.0	3.277
	PAR-0043	21	67.90	37.8	0.0	31.5	3.233
	PAR-0030	75	269.00	157.5	0.3	112.5	3.587
	PAR-0003	168	551.70	307.70	0.0	252.0	3.284
	PAR-0022	116	380.20	211.10	0.1	174.0	3.278
	PAR-0004	87	285.70	159.30	0.0	130.5	3.284

	PAR-0045	64	207.2	114.4	0.0	96.0	3.238
	PAR-0044	61	200.80	111.80	0.0	91.5	3.292
	PAR-0032	153	493.50	270.90	0.0	229.5	3.224
	PAR-0015	33	107.9	59.50	0.0	49.5	3.270

Table 2. Energy Consumption of Each Algorithm for Top 10 models

Name Of Algorithm	Keys	Avg. Accuracy	Avg. Time Taken(s)	Avg. Energy Used(J)	Avg. Equivalent CO_2 Emission(mg)
Bat Algorithm	['BAT-0042', 'BAT-0034', 'BAT-0008', 'BAT-0029', 'BAT-0011', 'BAT-0017', 'BAT-0003', 'BAT-0040', 'BAT-0016', 'BAT-0020']	93.13047	239.0	792.46	187.10861
Camel Algorithm	['CAM-0000', 'CAM-0049', 'CAM-0024', 'CAM-0003', 'CAM-0026', 'CAM-0008', 'CAM-0041', 'CAM-0033', 'CAM-0025', 'CAM-0006']	93.12125	242.5	779.44	184.03444
Cuckoo Search	['CUC-0009', 'CUC-0031', 'CUC-0010', 'CUC-0042', 'CUC-0035', 'CUC-0040', 'CUC-0041', 'CUC-0039', 'CUC-0046', 'CUC-0048']	93.07515	216.8	700.11	165.30375
Firefly Algorithm	['FIR-0042', 'FIR-0037', 'FIR-0003', 'FIR-0049', 'FIR-0043', 'FIR-0041', 'FIR-0014', 'FIR-0009', 'FIR-0012', 'FIR-0000']	93.14431	261.1	827.82	195.45750
Particle Swarm Optimization	['PAR-0016', 'PAR-0043', 'PAR-0030', 'PAR-0003', 'PAR-0022', 'PAR-0004', 'PAR-0045', 'PAR-0044', 'PAR-0032', 'PAR-0015']	93.12125	96.4	317.35	74.92986

Table 3. Average Accuracy and Energy Consumption for Each Algorithm

Fig. 3 compares the energy usage of each method for clarity, and Fig. 4 displays the average accuracy of the five algorithms for the top 10 epochs.

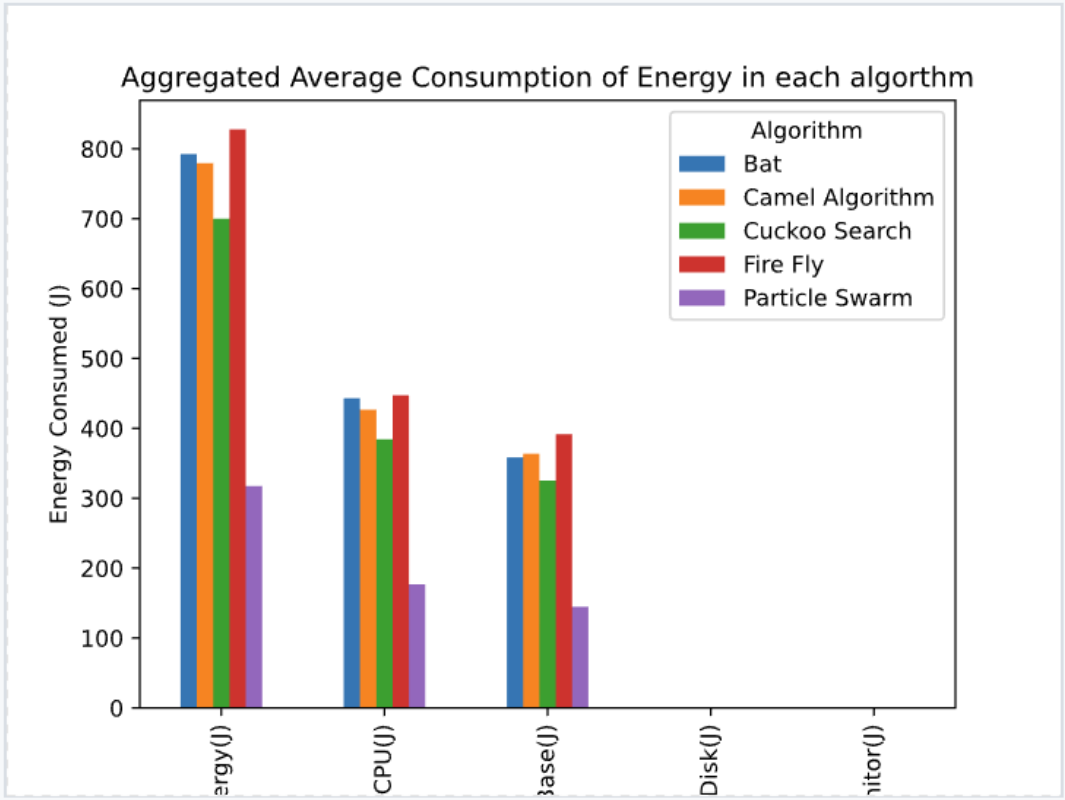

Figure 3. Aggregated Average consumption of Energy in each Algorithm

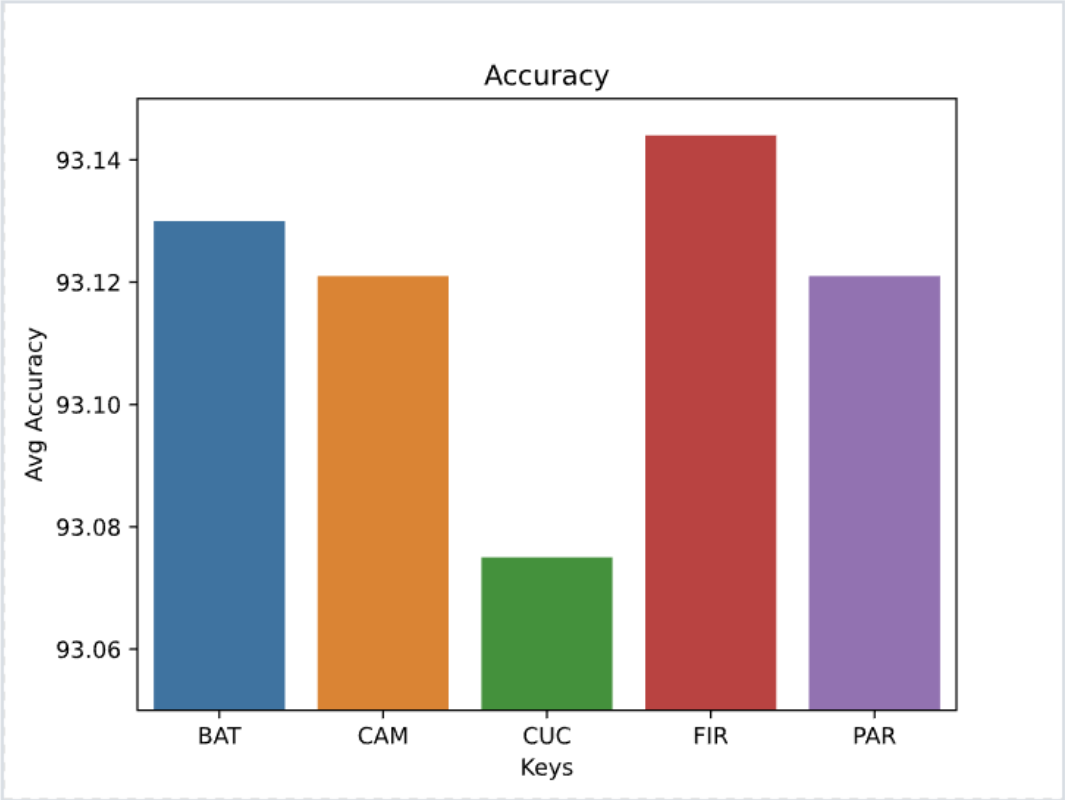

Figure 4. Average Accuracy of top 10 epochs achieved for each of the five nature-inspired algorithms

Fig. 5 shows the average energy consumed by the five algorithms for top 10 epochs, while Fig. 6 shows the average CO_2 emitted by the five algorithms for top 10 epochs.

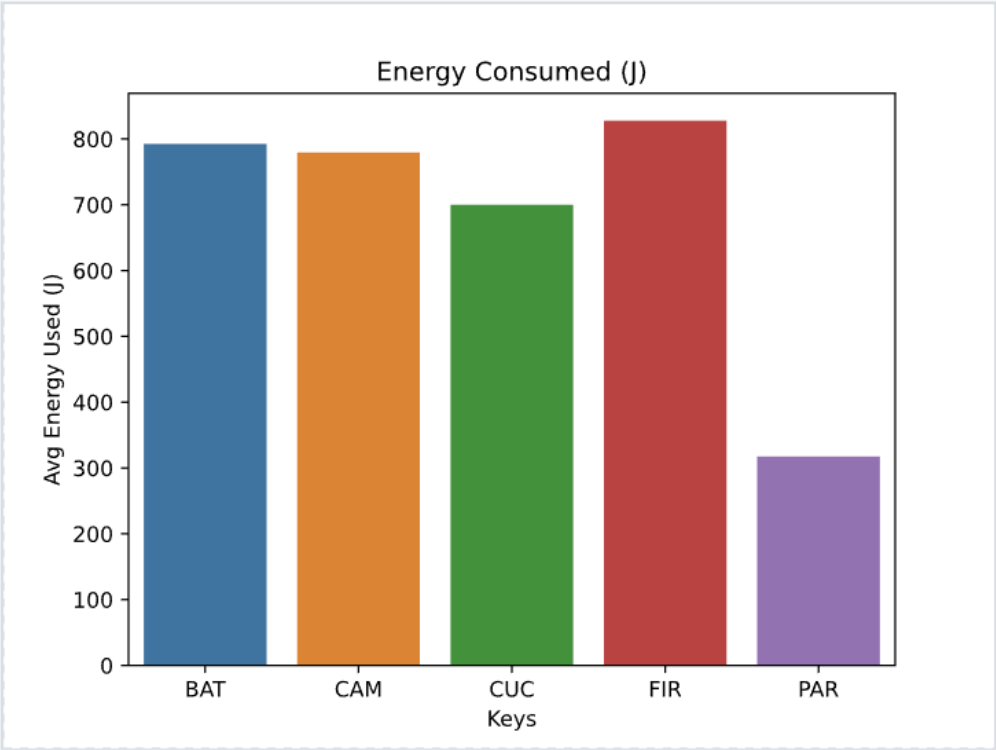

Figure 5. Average Energy Consumed of top 10 epochs achieved for each of the five nature-inspired algorithms

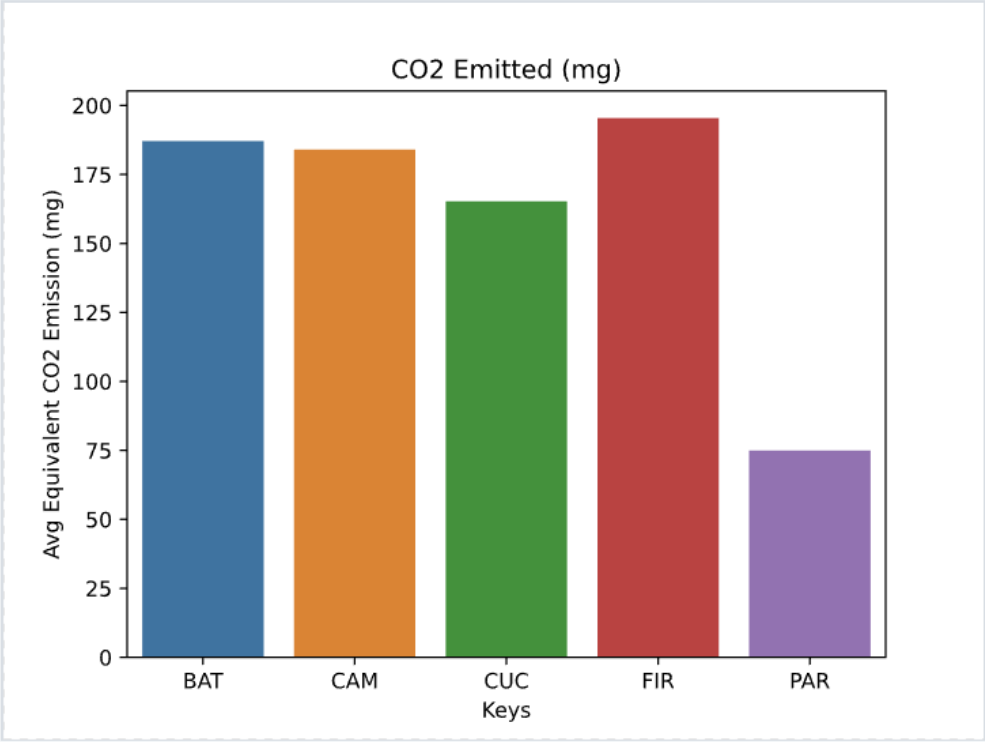

Figure 6. Average Equivalent CO_2 emissions of top 10 epochs achieved for each of the five nature-inspired algorithms

As was previously said, different hardware specs would produce various outcomes. As a result, the outcomes will alter if the trials are carried out on a laptop with different specifications. PAR is used as the foundation to explore the energy consumption ratio of other algorithms, which is presented in Table 4, because it is discovered to have the lowest energy consumption. .

Algorithm	CPU Energy Consumption(J)	Ratio Comparison to PAR
Bat Algorithm	443.18	2.505
Camel Algorithm	426.63	2.412
Cuckoo Search	384.29	2.172
Firefly Algorithm	447.29	2.529
Particle Swarm	176.89	1.0
Algorithm	Total Energy Consumed(J)	Ratio Comparison to PAR
Bat Algorithm	792.46	2.497
Camel Algorithm	779.44	2.456
Cuckoo Search	700.11	2.206
Firefly Algorithm	827.82	2.609
Particle Swarm	317.35	1.0

Table 4. Energy Consumption Ratio for Each Algorithm

Energy usage for each optimization algorithm varies greatly. But as the number of decision trees increases, it is observed that Particle Swarm Optimization algorithm has the highest accuracy to energy consumption ratio of 0.29343. Firefly Algorithm performs the worst with the accuracy to energy consumption ratio of 0.11252 (Fig 7).

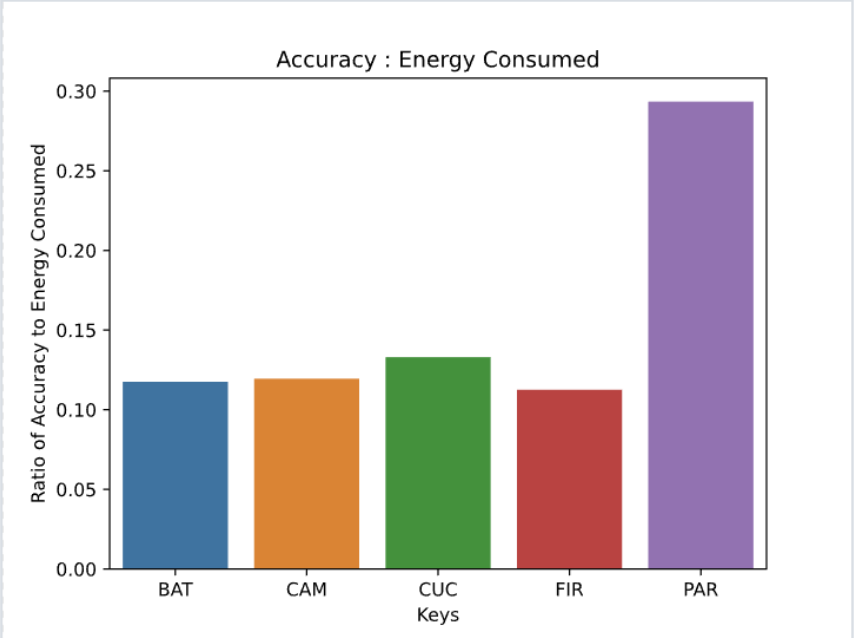

Figure 7. Ratio of Accuracy to Energy Consumption of top 10 epochs achieved for each of the five nature-inspired algorithm

5. Discussion:

Despite their widespread use and efficiency, NIO algorithms have a few difficult issues. Every NIO method has algorithm-dependent parameters, and these parameters' values can greatly impact how well the algorithm performs. It is presently unknown what the proper value of these parameters should be to achieve an ideal balance between exploration and exploitation for a particular algorithm and a certain set of problems because parameter selections may change based on the algorithm or concerns. As a result, it is feasible to research how parameter values for NIO algorithms can be adjusted and regulated to enhance performance while reducing energy consumption.

To explore for potential relationships between NIO approaches and the energy consumption of hardware resources, it is possible to compile statistics on the energy usage of hardware resources across a variety of CPU architectures. These projects will all shed light on the energy effectiveness of NIO algorithms for complex applications.

6. Additional Requirements

Nature inspired algorithms implementation provided by the NiaPy machine learning library.

7. References

[1] Li, H., Liu, X., Huang, Z., Zeng, C., Zou, P., Chu, Z. and Yi, J., 2020. Newly emerging nature-inspired optimization-algorithm review, unified framework, evaluation, and behavioural parameter optimization. *IEEE Access*, *8*, pp.72620-72649.https://ieeexplore.ieee.org/abstract/document/9064786/

[2] Yang X-S. Nature-inspired Optimization Algorithms. London: Academic Press; 2020.

[3] Bayer H, Nebel M. Evaluating algorithms according to their energy consumption. Math Theory Comput Pract. 2009;48:1–25.

[4] Podder S, Burden A, Singh SK, Maruca R. Sustainable Business practices – How Green Is Your Software? 2020. Harvard Business Review. https://hbr. org/2020/09/how-green-is-your-software Accessed 8 Jan 2022.

[5] Heaton, J. Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning. Genet Program Evolvable Mach **19**, 305–307 (2018). https://doi.org/10.1007/s10710-017-9314-z.

[6] Schwartz, R., Dodge, J., Smith, N., & Etzioni, O. (2019). Green AI. Communications of the ACM, 63, 54 - 63.

[7] X.-S. Yang, A New Metaheuristic Bat-Inspired Algorithm, in: Nature Inspired Cooperative Strategies for Optimization (NICSO 2010) (Eds. J. R. Gonzalez et al.), SCI 284, 65-74 (2010) https://doi.org/10.48550/arXiv.1004.4170

[8] Gandomi, A.H., Yang, XS., Alavi, A.H. et al. Bat algorithm for constrained optimization tasks. Neural Comput & Applic 22, 1239–1255 (2013). https://doi.org/10.1007/s00521-012-1028-9

[9] Ibrahim, M.K., & Ali, R.S. (2016). Novel Optimization Algorithm Inspired by Camel Traveling Behavior. Journal of Electrical and Electronic Engineering, 12, 167-177.

[10] X.-S. Yang; S. Deb (December 2009). Cuckoo search via Lévy flights. World Congress on Nature & Biologically Inspired Computing (NaBIC 2009). IEEE Publications. pp. 210–214. https://doi.org/10.48550/arXiv.1003.1594

[11] Gandomi, A.H., Yang, XS. & Alavi, A.H. Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Engineering with Computers 29, 17–35 (2013). https://doi.org/10.1007/s00366-011-0241-y

[12] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360), 1998, pp. 69-73, doi: 10.1109/ICEC.1998.699146.

[13] Sheta AF, Ayesh A, Rine D. Evaluating software cost estimation models using particle swarm optimisation and fuzzy logic for nasa projects: a comparative study. Int J Bio-Inspired Comput. 2010;2(6):365–73.

[14] Saini S, Bt Awang Rambli DR, Zakaria MNB, Bt Sulaiman S. A review on particle swarm optimization algorithm and its variants to human motion tracking. Math Probl Eng. 2014;2014:1–16.

[15] Mohana R. A position balanced parallel particle swarm optimization method for resource allocation in cloud. Indian J Sci Technol. 2015;8(S3): 182–8

[16] Delice Y, Kızılkaya Aydogan E, Özcan U, undefinedlkay MS. A modified ˘ particle swarm optimization algorithm to mixed-model two-sided assembly line balancing. J Intell Manuf. 2017;28(1):23–36.

[17] Esmin AA, Coelho RA, Matwin S. A review on particle swarm optimization algorithm and its variants to clustering high-dimensional data. Artif Intell Rev. 2015;44(1):23–45.

[18] Li M, Du W, Nian F. An adaptive particle swarm optimization algorithm based on directed weighted complex network. Math Probl Eng. 2014;2014:1–7

[19] Pereira R, Couto M, Ribeiro F, Rua R, Cunha J, Fernandes JP, Saraiva J. Energy efficiency across programming languages: how do energy, time, and memory relate? In: Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering. New York: Association for Computing Machinery; 2017. p. 256–67.

[20] Georgiou S, Kechagia M, Spinellis D. Analyzing programming languages' energy consumption: An empirical study. In: Proceedings of the 21st Pan-Hellenic Conference on Informatics. New York: Association for Computing Machinery; 2017. p. 1–6.

[21] Pereira R, Couto M, Ribeiro F, Rua R, Cunha J, Fernandes JP, Saraiva J. Ranking programming languages by energy efficiency. Sci Comput Program. 2021;205:102609.

[22] Rashid M, Ardito L, Torchiano M. Energy consumption analysis of algorithms implementations. In: 2015 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM). Beijing: IEEE; 2015. p. 1–4

[23] Verma M, Chowdhary K. Analysis of energy consumption of sorting algorithms on smartphones. In: Proceedings of 3rd International Conference on Internet of Things and Connected Technologies (ICIoTCT). Rochester: ELSEVIER-SSRN; 2018. p. 472–5.

[24] Deepthi T, Birunda A. Time and energy efficiency: A comparative study of sorting algorithms implemented in c. In: International Conference on Advancements in Computing Technologies-ICACT 2018. vol. 4. India: IJFRCSCE; 2018. p. 25–7.

[25] Ayodele OS, Oluwade B. A Comparative Analysis of Quick, Merge and Insertion Sort Algorithms using Three Programming Languages II: Energy Consumption Analysis. Afr J MIS. 2019;1(2):44–63.

[26] Jamil, M., Kor, AL. Analyzing energy consumption of nature-inspired optimization algorithms. GRN TECH RES SUSTAIN 2, 1 (2022). https://doi.org/10.1007/s44173-021-00001-9

[27] Charlotte Freitag, Mike Berners-Lee, Kelly Widdicks, Bran Knowles, Gordon S. Blair, Adrian Friday, The real climate and transformative impact of ICT: A critique of estimates, trends, and regulations, Patterns, Volume 2, Issue 9, 2021, 100340, ISSN 2666-3899, https://doi.org/10.1016/j.patter.2021.100340

[28] Murugesan S. Going Green with IT: Your Responsibility Toward Environmental Sustainability. Arlington: Cutter Consortium; 2007

[29] Simunic T, Benini L, De Micheli G. Energy-efficient design of battery-powered embedded systems. IEEE Trans Very Large Scale Integr (VLSI) Syst. 2001;9(1):15–28.

[30] Schmitz MT, Al-Hashimi BM, Eles P. System-level Design Techniques for Energy-efficient Embedded Systems. Berlin: Springer Science & Business Media; 2004.

[31] Hosangadi A, Kastner R, Fallah F. Energy efficient hardware synthesis of polynomial expressions. In: 18th International Conference on VLSI Design Held Jointly with 4th International Conference on Embedded Systems Design. India: IEEE; 2005. p. 653–8.

[32] Shiri A, Mazumder AN, Prakash B, Manjunath NK, Homayoun H, Sasan A, Waytowich NR, Mohsenin T. Energy-efficient hardware for language guided reinforcement learning. In: Proceedings of the 2020 on Great Lakes Symposium on VLSI. New York: Association for Computing Machinery; 2020. p. 131–6.

[33] Capra E, Francalanci C, Slaughter SA. Is software “green”? Application development environments and energy efficiency in open source applications. Inf Softw Technol. 2012;54(1):60–71.

[34] D’Agostino D, Merelli I, Aldinucci M, Cesini D. Hardware and software solutions for energy-efficient computing in scientific programming. Sci Prog. 2021;2021:1–9.

[35] Naumann S, Dick M, Kern E, Johann T. The greensoft model: A reference model for green and sustainable software and its engineering. Sustain Comput Inf Syst. 2011;1(4):294–304.

[36] Kor A-L, Pattinson C, Imam I, AlSaleemi I, Omotosho O. Applications, energy consumption, and measurement. In: 2015 International Conference on Information and Digital Technologies. Zilina: IEEE; 2015. p. 161–171.

[37] Pattinson C, Olaoluwa PO, Kor A-L. A comparative study on the energy consumption of PHP single and double quotes. In: 2015 IEEE International Conference on Data Science and Data Intensive Systems. Sydney: IEEE; 2015. p. 232–9.

[38] Engel M. Sustainable software design. In: Green Information Technology. San Francisco: Elsevier; 2015. p. 111–27.

[39] Dastbaz M, Pattinson C, Akhgar B. Green Information Technology: A Sustainable Approach. San Francisco: Morgan Kaufmann; 2015

[40] Ardito L, Procaccianti G, Torchiano M, Vetro A. Understanding green software development: A conceptual framework. IT Prof. 2015;17(1):44–50.

[41] Murugesan S. Harnessing green IT: Principles and practices. IT Prof. 2008;10(1):24–33.

[42] Ferreira MA, Hoekstra E, Merkus B, Visser B, Visser J. Seflab: A lab for measuring software energy footprints. In: 2013 2nd International Workshop on Green and Sustainable Software (GREENS). San Francisco: IEEE; 2013. p. 30–7.

[43] Bener AB, Morisio M, Miranskyy A. Green software. IEEE Softw. 2014;31(3): 36–9

[44] Barontini A, Masciotta M-G, Ramos LF, Amado-Mendes P, Lourenço PB. An overview on nature-inspired optimization algorithms for structural health monitoring of historical buildings. Proc Eng. 2017;199:3320–5. https://doi.org/10.1016/j.proeng.2017.09.439. X International Conference on Structural Dynamics, EURODYN 2017.

[45] Yang X-S. Nature-inspired optimization algorithms: Challenges and open problems. J Comput Sci. 2020;46:101104.

[46] Abdollahzadeh B, Soleimanian Gharehchopogh F, Mirjalili S. Artificial gorilla troops optimizer: A new nature-inspired metaheuristic algorithm for global optimization problems. Int J Intell Syst. 2021;36(10):5887–958. https://doi.org/10.1002/int.22535.

[47] Mohanty A, Nag KS, Bagal DK, Barua A, Jeet S, Mahapatra SS, Cherkia H. Parametric optimization of parameters affecting dimension precision of fdm printed part using hybrid taguchi-marcos-nature inspired heuristic optimization technique. Mater Today Proc. 2021. https://doi.org/10.1016/j.matpr.2021.06.216.

[48] Sadrnia A, Soltani HR, Zulkifli N, Ismail N, Ariffin MKA. A review of nature-based algorithms applications in green supply chain problems. Int J Eng Technol. 2014;6(3):204–11.

[49] Nguyen T-H, Nguyen LV, Jung JJ, Agbehadji IE, Frimpong SO, Millham RC. Bio-inspired approaches for smart energy management: State of the art and challenges. Sustainability. 2020;12(20):. https://doi.org/10.3390/su12208495 .

[50] Usman MJ, Ismail AS, Abdul-Salaam G, Chizari H, Kaiwartya O, Gital AY, Abdullahi M, Aliyu A, Dishing SI. Energy-efficient nature-inspired techniques in cloud computing datacenters. Telecommun Syst. 2020;71: 275–302. https://doi.org/10.1007/s11235-019-00549-9 .

[51] Sharma R, Vashisht V, Singh U. Nature inspired algorithms for energy efficient clustering in wireless sensor networks. In: 2019 9th International Conference on Cloud Computing, Data Science Engineering (Confluence); 2019. p. 365–70. https://doi.org/10.1109/CONFLUENCE.2019.8776618 .

[52] Agbehadji IE, Millham RC, Abayomi A, Jung JJ, Fong SJ, Frimpong SO. Clustering algorithm based on nature-inspired approach for energy optimization in heterogeneous wireless sensor network. Appl Soft Comput. 2021;104:107171. https://doi.org/10.1016/j.asoc.2021.107171 .

[53] Johnson, B. A., & Iizuka, K. (2016). Integrating OpenStreetMap crowdsourced data and Landsat timeseries imagery for rapid land use/land cover (LULC) mapping: Case study of the Laguna de Bay area of the Philippines. Applied Geography, 67, 140-149.

[54] Sklearn-nature-inspired-algorithm https://pypi.org/project/sklearn-nature-inspired-algorithms/

[55] NiaPy https://niapy.org/en/stable/

[56] Kansal A, Goraczko M, Liu J, Zhao F. Joulemeter: Computational Energy Measurement and Optimization;2010. Microsoft Research, Redmond, United States. https://www.microsoft.com/en-us/research/project/joulemeter-computational-energy-measurement-and-optimization/ Accessed 8 Jan 2022.

[57] Central Electricity Authority of India Guidelines for energy to CO2 emissions: https://cea.nic.in/wp-content/uploads/baseline/2020/07/user_guide_ver14.pdf

8. Supplementary Materials:

The supplementary figures and tables are available in separate file.

9. Funding:

No external funding has been utilized for conducting the study.

10. Data Availability Statements:

The dataset used is Crowdsourced data from OpenStreetMap is used to automate the classification of satellite images into different land cover classes (impervious, farm, forest, grass, orchard, water) [57]. All the data generated by 5 NIOAs, the results are included within the manuscript and supplementary materials.

11. Acknowledgements:

The authors and contributors acknowledge active support from their college Maharaja Agrasen Institute of Technology, Rohini, New Delhi.

12. Conflicts of Interest:

The authors declare no conflict of interest.