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: Machine learning can be said to be the key to analyzing data for decision-making. With this much usage, it becomes important that these algorithms run under minimum resources so we can reduce recurring costs and provide efficient results in less time. An optimizer is a method or algorithm to update the various parameters that can reduce the loss with much less effort. One such optimizer is the Nature-Inspired Optimization (NIO) algorithm. They are highly efficient in finding optimized solutions to multi-dimensional and multi-modal problems. Over previous decades, many nature-inspired optimization algorithms (NIOAs) have been proposed and applied due to their importance and significance.

This study presents a critical analysis of energy consumption and the corresponding carbon footprint for some popular NIO algorithms. Microsoft Joulemeter is employed for measuring the energy consumption during the runtime of each algorithm. In contrast, the corresponding carbon footprint of each algorithm is calculated based on India's Central Electricity Authority guide. The results of this study evidence that each algorithm demonstrates different energy consumption behaviors to achieve the same goal. This study will help software developers to choose better (greener) options among the tested NIO algorithms. Future research can take into account additional NIO algorithms and their variations for energy consumption analysis to determine the most environmentally friendly NIO algorithms. Furthermore, additional work might be considered to determine the potential impact of alternative CPU architectures on the performance and energy consumption of the NIO algorithms.

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

1. Introduction

Broadly speaking, optimization is the act of changing an existing process in order to increase the occurrence of favorable outcomes and decrease the occurrence of unfavorable outcomes. Optimization is a common mathematical problem in various fields such as engineering, business operations, industrial designs, etc. 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.

Computing energy usage should be considered while designing programs targeting elevated performance and mobile software applications due to the rise of mobile and IoT devices. Software programs' improved algorithms & data structures can make them more environmentally and energy-friendly. In previous decades, the sole performance indicator taken into account for study as well as optimization of an algorithm was runtime, which served as the basis for evaluating an algorithm's performance [3]. But the rapid development of high-performance computers and embedded systems with faster processors in recent years has led to a rise in energy usage. As a result, it is essential to take into account an algorithm's energy consumption while evaluating it (i.e., in terms of performance and sustainability). Since an algorithm's implementation will affect energy usage and environmental impact, its efficacy and efficiency must be evaluated in the context of a specific application.

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]. However, this work intends to examine the energy consumption and accompanying carbon footprint for some commonly used NIO algorithms, namely Bat Algorithm (BAT), Camel Algorithm (CAM), Cuckoo Search (CS), Firefly Algorithm (FIR) and Particle Swarm Optimization (PAR). Due to the vast range of applications for these algorithms, they were taken into consideration for this study. This study intends to show how one may experimentally evaluate the energy usage of various algorithms. Keep in mind that future work could focus on alternative NIO algorithms.

Bat Algorithm (BAT): 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): A novel optimization algorithm inspired by the traveling behavior of Camel in the desert in difficult environments. A Camel tends to move towards a region that contains food and water. Under that consideration, several factors and operators are considered to outline CA algorithm procedure, including temperature effect, The supply (water and food), The camel endurance, Camel visibility (and /or hearing) range, Random walk, Group effect (multi-solution), Termination condition (dying or moving back), 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 (CS): Cuckoo search is an optimization algorithm developed by Xin-She Yang and Suash Deb in 2009 [10]. It was inspired by the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of host birds of other species. Some host birds can engage direct conflict with the intruding cuckoos. For example, if a host bird discovers the eggs are not their own, it will either throw these alien eggs away or simply abandon its nest and build a new nest elsewhere. Some cuckoo species such as the New World brood-parasitic Tapera have evolved in such a way that female parasitic cuckoos are often very specialized in the mimicry in colors and pattern of the eggs of a few chosen host species. Cuckoo search idealized such breeding behavior, and thus can be applied for various optimization problems [11].

Firefly Algorithm (FIR): 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): Particle Swarm Optimization (PSO) is derived from the swarm intelligence of flocking of birds and schooling of fishes in search of food, where each particle contains its own velocity and position [12]. It has been used to address several problems such as software cost estimation [13], human motion tracking [14], resource allocation in the cloud [15], assembly line balancing [16], data clustering [17], etc. However, a limitation of PSO is it easily falls into local optimum in high-dimensional space and has a low convergence rate in the iterative process [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.

Several existing work has conducted a comparative analysis of the energy consumption of different programming languages [19–21] and sorting algorithms implementations [22–25]. Energy consumption and greenness of NIO algorithms has only be measured by one other teams. They measured the energy efficiency of Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Differential Evolution (DE) and Artificial Bee Colony (ABC) [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 remainder of this article is organized as follows: a literature review on the environmental impact of ICT, green or energy-efficient software, the influence of hardware energy consumption by software, and related works on the analysis of energy consumption in algorithms implementations as well as nature-inspired algorithms and energy efficiency is presented in Section 2. Section 3 provides a brief overview of methodologies, which include both macro and micro methodology as well as experiment setup and design. Findings and discussion are presented in Section 4, which discusses energy consumption and corresponding carbon footprint of each algorithm, and ethical issues and challenges of this study. A summary of the discussion and recommendations for future studies is presented in Section 5

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]. Because global demand for ICT products and services is expanding, the ICT sector can play a critical role in lowering global carbon emissions by reducing the carbon footprints of its products and services. Though much research has been conducted to make hardware and other embedded systems more energy-efficient [29-32], a similar emphasis should be placed on the creation of energy-efficient software and applications [33, 34].

a. Energy-efficient software

When a piece of software uses less energy for its effective computing and does little environmental damage, it is referred to as green or energy-efficient [35]. Several studies have been undertaken on the energy efficiency of web-based software applications [36] and software features [37]. The direct impact of software on a laptop or mobile battery is easily quantified as 25% to 40% of overall energy utilized by a device [38]. However, the indirect influence of software is more difficult to quantify because it is linked to the life cycle of the host device [39]. Energy-efficiency of a software can only be truly achieved only when both the positive and negative impacts are properly taken into account throughout the design and deployment phases. In light of this, ICT application service optimization is crucial to reducing negative environmental effects.

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

Software's hardware-related energy usage habits directly affect how much energy hardware uses and how long a device's battery lasts [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]. The creation of energy-efficient software that improves a piece of hardware's energy efficiency is one of the trickiest challenges during the design phase of an embedded system. In order to make software and applications more productive while still being energy-efficient, various trade-offs between performance and sustainability will need to be taken into account [43].

c. Analysis of Energy Consumption in algorithms implementations

In a study, four sorting algorithms, namely Bubble, Merge, Quick, and Counting sort, have been examined for their energy efficiency by Rashid and colleagues [22]. An experiment was set up on an ARM-based device, and it was determined how much energy was used by four sorting algorithms written in three different programming languages. According to this investigation, the Counting sort implementation in ARM assembly language was the most environmentally friendly choice.

Five sorting algorithms—Bubble, Insertion, Quick, Selection, and Counting sort—have had their energy consumption measured in [23]. To measure energy consumption in this study, five separate Apps were created, one for each sorting method. According to this study, Bubble sort is the most energy-intensive algorithm, whereas Quick sort is the most energy-efficient sorting technique in typical situations.

Deepthi and colleagues have conducted experiments to study how different sorting algorithms have an impact on energy consumption using C language implementation [24]. This study found that both time and energy have an impact on the efficiency of these sorting algorithms. This study considered six sorting algorithms, namely Quick, Merge, Shell, Insertion, Selection, and Bubble sort. It has been found that the energy consumption of Quick, Merge, and Shell sort is similar while Insertion and Selection sorts are far better than Bubble sort.

Utilizing three programming languages (C, Java, and Python), two algorithm implementation styles (Iterative and Recursive), and three algorithm types (Quick, Merge, and Insertion), Ayodele and colleagues conducted a comparative experimental analysis of the energy consumption of these three algorithms [25]. According to this study, the amount of energy consumed depends on the size of the data, the programming language used, and the way the algorithms are implemented. Additionally, in order to reduce energy consumption, this study offers guidance for selecting the sorting algorithm type and its algorithm implementation style.

In research conducted by Jamil and Kor, energy consumption of a few nature-inspired algorithms has been analyzed on a dataset [26]. The algorithms used were Genetic Algorithm, Particle Swarm Optimization, Differential Evolution and Artificial Bee Colony algorithm. Each optimization algorithm exhibits significantly different energy consumption, where Differential Evolution (DE) is found to be greenest compared to other algorithms.

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. Our critical literature review has shown that to date, there are few studies on energy-efficient nature-inspired algorithms and thus, our research aims to address to further promote study in our area.

3. Methodology

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

- Micro methodology

a. Data Collection:

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 [53] and NiaPy, a library dedicated to Nature Inspired Optimization Algorithm in Python [54].

b. Energy Profiling:

The estimated energy consumption of each NIO method was calculated using Microsoft Joulemeter software [55], which can track the energy used by a running program or software as well as by particular hardware resources, including CPU, Monitor, Disk, and Idle or Base power.

c. Carbon Footprint:

The guidelines of the Central Electricity Authority of India have been used as method for calculating carbon emissions [56]. After obtaining the Energy consumed for an experiment (in terms of kWh), the data is converted to equivalent carbon emitted based on the following formula,

$$\text{CO2 Emissions} = 0.85 * E(\text{kW-hr/year}) \text{ where E is the energy consumed.}$$

$$1\text{kWhr of Energy Consumed} = 0.85\text{Kg of } CO_2 \text{ emission}$$

$$72 \text{ Joules} = 17 \text{ mg of } CO_2 \text{ emissions}$$

- **Experiment Setup**

a. System Specification:

Different hardware specifications would bring about different results. Therefore, all experiments were conducted on a laptop with the following specifications shown

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.

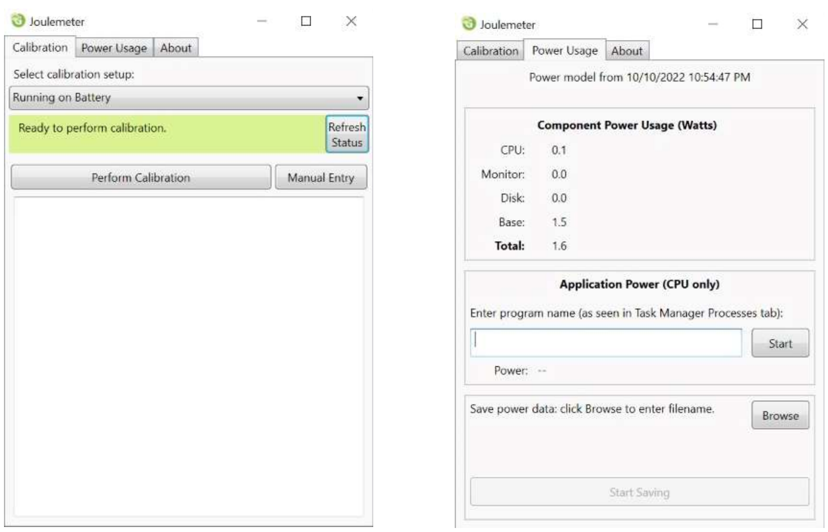

Figure. 1 Calibration of Joulemeter

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.

The corresponding result of each algorithm has been stored in a separate CSV file. All results of each algorithm have been aggregated in an Excel file for analyzing the energy consumption of these five algorithms. The design of experiments can be summarized as follows.

- Nature-Inspired Optimization (NIO) Algorithms: {PSO, CS, CAM, BAT, FF}
- Programming Language: Python
- Benchmark Function: Sphere Function
- Search Space: [-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

For better illustration, Fig. 2 shows a comparison of energy consumption for each algorithm, while Fig. 3 shows the average accuracy of the five algorithms for top 10 epochs.

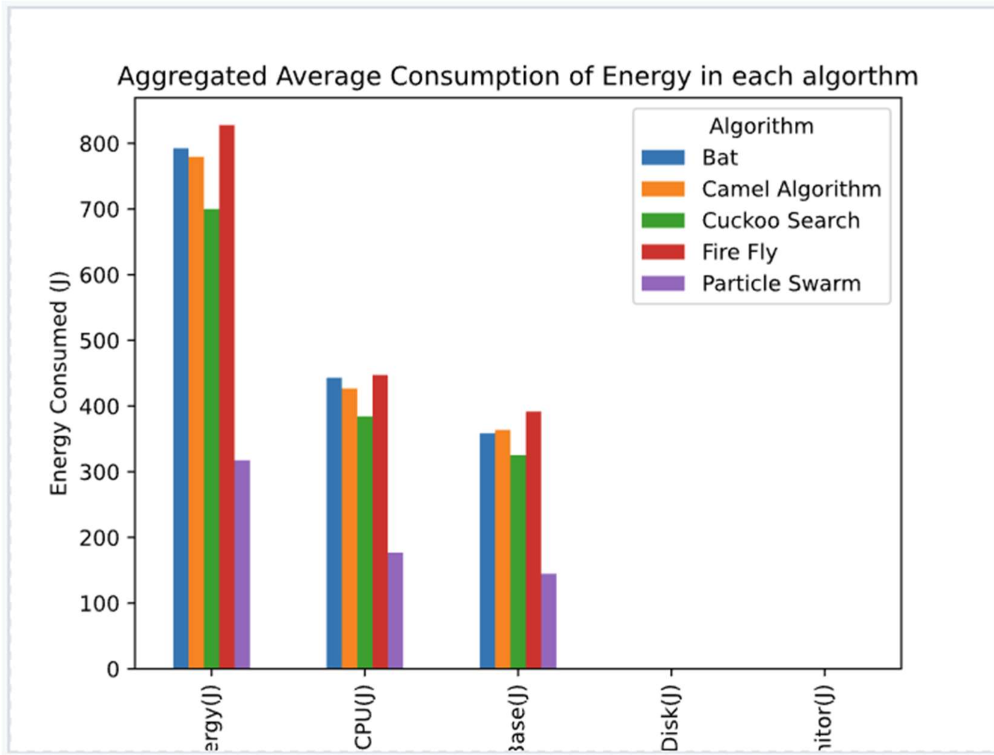

Figure 2. Aggregated Average consumption of Energy in each Algorithm

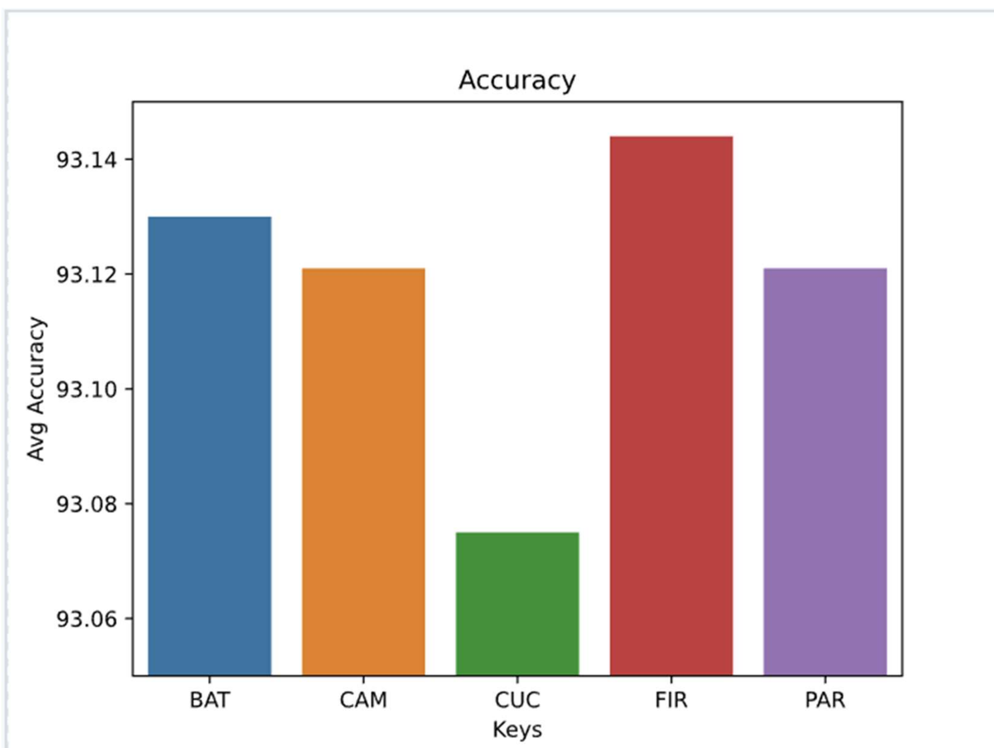

Figure 3. Average Accuracy of top 10 epochs achieved for each of the five nature-inspired algorithms

Fig. 4 shows the average energy consumed by the five algorithms for top 10 epochs, while Fig. 5 shows the average CO_2 emitted by the five algorithms for top 10 epochs.

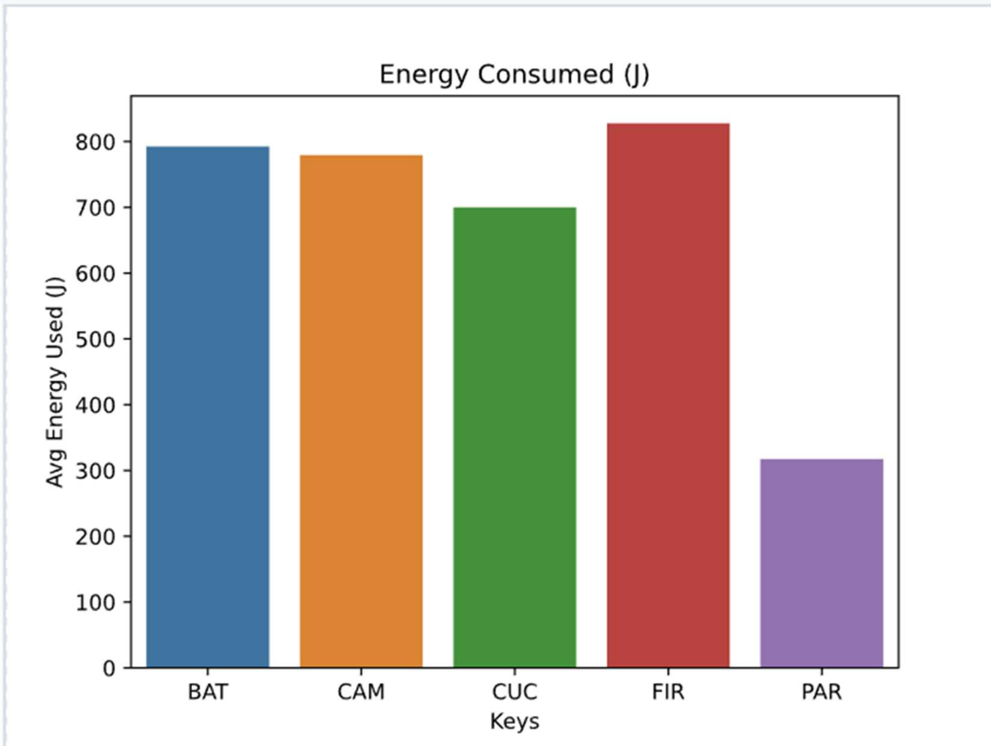

Figure 4. Average Energy Consumed of top 10 epochs achieved for each of the five nature-inspired algorithms

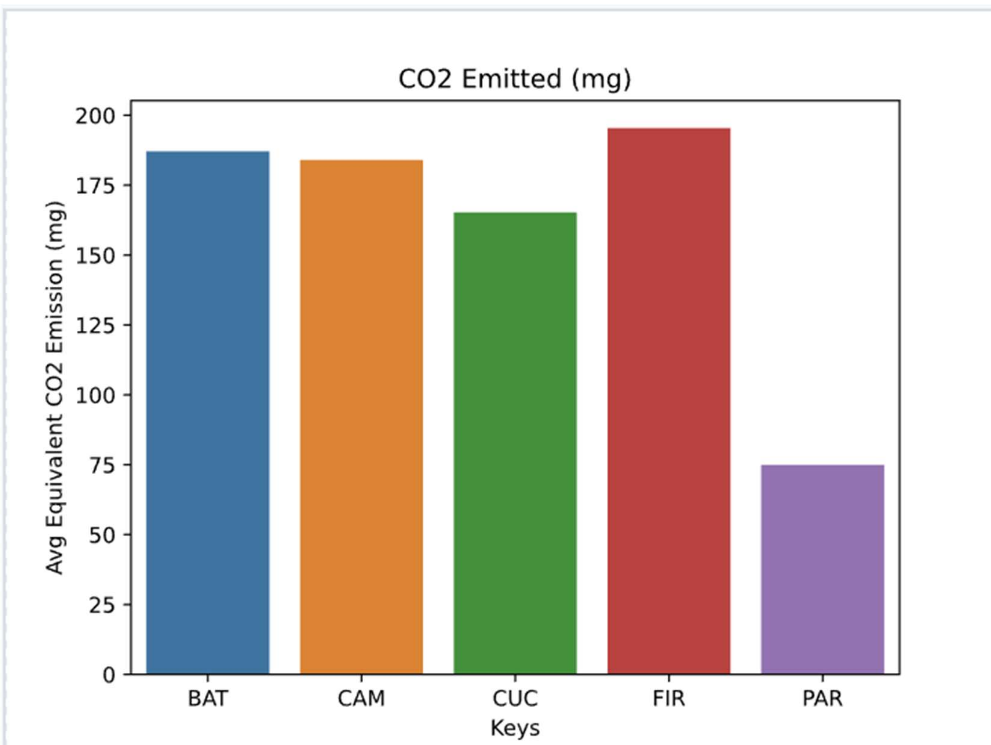

Figure 5. Average Equivalent CO_2 emissions of top 10 epochs achieved for each of the five nature-inspired algorithms

As mentioned earlier, different hardware specifications would bring about different results. Therefore, if the experiments are conducted on a laptop with different specifications, the result will vary. Since PSO is found to have the lowest energy consumption, it is used as the base to investigate the energy consumption ratio of other algorithms which is shown in Table 4.

Algorithm	CPU Energy Consumption(J)	Ratio Comparison to PSO
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 PSO
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 6).

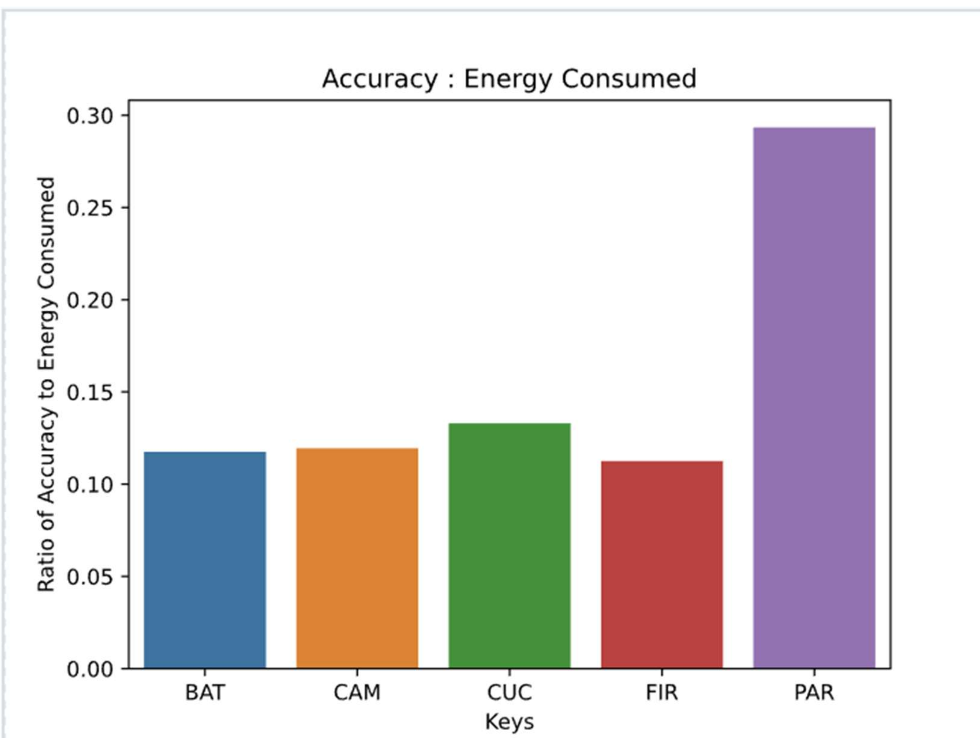

Figure 6. Ratio of Accuracy to Energy Consumption of top 10 epochs achieved for each of the five nature-inspired algorithm

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. As parameter choices might vary depending on the algorithm or issues, it is currently unclear what the appropriate value of these parameters is to achieve an optimal balance between exploration and exploitation for a specific algorithm and a given collection of problems. As a result, it is possible to investigate the tweaking and regulating of NIO algorithms' parameter values to improve their performance while minimizing their energy usage.

It is feasible to gather data on the energy consumption of hardware resources across a range of CPU architectures to look for potential connections between NIO techniques and the energy consumption of hardware resources. All of these initiatives will provide insight into the energy efficiency of NIO algorithms for sophisticated applications.

6. Additional Requirements

Nature inspired algorithms implementation provided by the scikit-learn 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] Sklearn-nature-inspired-algorithm https://pypi.org/project/sklearn-nature-inspired-algorithms/

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

[55] 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.

[56] 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

[57] 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.

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.