Contents lists available at ScienceDirect

Array

journal homepage: www.elsevier.com/locate/array

A comprehensive framework for hand gesture recognition using hybrid-metaheuristic algorithms and deep learning models

Hassan Mohyuddin[a], Syed Kumayl Raza Moosavi[b], Muhammad Hamza Zafar[c], Filippo Sanfilippo[c,d,*]

[a] Department of Electrical Engineering, Capital University of Sciences and Technology, Islamabad 44000, Pakistan
[b] School of Electrical Engineering and Computer Science, National University of Sciences and Technology, Islamabad 44000, Pakistan
[c] Department of Engineering Sciences, University of Agder, Grimstad 4879, Norway
[d] Department of Software Engineering, Kaunas University of Technology, Kaunas 51368, Lithuania

ARTICLE INFO

Keywords:
Feature selection
Dimensional reduction
Big data
Meta heuristic algorithms
Machine learning

ABSTRACT

This paper presents a novel methodology that utilizes gesture recognition data, which are collected with a Leap Motion Controller (LMC), in tandem with the Spotted Hyena-based Chimp Optimization Algorithm (SSC) for feature selection and training of deep neural networks (DNNs). An expansive tabular database was created using the LMC for eight distinct gestures and the SSC algorithm was used for discerning and selecting salient features. This refined feature subset is then utilized in the subsequent training of a DNN model. A comprehensive comparative analysis is conducted to evaluate the performance of the SSC algorithm in comparison with established optimization techniques, such as Particle Swarm Optimization(PSO), Grey Wolf Optimizer(GWO), and Sine Cosine Algorithm(SCA), specifically in the context of feature selection. The empirical findings decisively establish the efficacy of the SSC algorithm, consistently achieving a high accuracy rate of 98% in the domain of gesture recognition tasks. The feature selection approach proposed emphasizes its intrinsic capacity to enhance not only the accuracy of gesture recognition systems and its wider suitability across diverse domains that require sophisticated feature extraction techniques.

1. Introduction

Big data analytics is one of the key areas where the next generation of parallel and distributed systems will be used. Nowadays, exabyte-sized data repositories for these applications exist and are growing quickly. These datasets, in addition to their sheer size, provide major obstacles to techniques and software development. The volume and security concerns of datasets, which are frequently dispersed, call for distributed methods using platforms used to store data frequently have a wide range of computing and networking capabilities. Many applications require careful consideration to fault-tolerance, security, and access control [1]. Analysis tasks often come with tight deadlines, and in certain cases, ensuring high-quality data is a significant challenge. Data-driven models and techniques that can function at scale are yet unknown for most developing applications.

The term “Big Data” encompasses not only the immense volume of data but also its intricate complexity and diverse variety [2], and velocity [3]. The advent of big data has revolutionized organizational operations and decision-making processes [4]. Through the utilization of advanced big data analytical tools and technologies, organizations can harness valuable insights into customer behavior, market trends, and operational performance. These insights enable informed decision-making, drive efficiency enhancements, and foster innovation within the organization [4]. But the sheer volume of data alone cannot better systems, it is not just the quantity of data but also its quality that affects the output of a system [5]. The presence of poor-quality data can have detrimental effects on decision-making and business operations, as it may result in inaccurate and unreliable outcomes. Such consequences can significantly impact the organization’s effectiveness and success. Therefore, ensuring data quality is of utmost importance to mitigate risks and make sound decisions based on trustworthy information [6]. Therefore, good quality data is an empirical part of modern technology and it is being used everywhere from using machine learning (ML) algorithms to redefine healthcare and make more accurate diagnoses [7] to its use in reshaping our modern-day financial decisions [8].

Human action recognition is also an area that has become increasingly popular in the pattern recognition and computer vision areas as a consequence of the expansion of numerous interactive applications in human–computer interaction. Hand gesture detection (HGD) is used

* Corresponding author at: Department of Engineering Sciences, University of Agder, Grimstad 4879, Norway.
E-mail address: filippo.sanfilippo@uia.no (F. Sanfilippo).

https://doi.org/10.1016/j.array.2023.100317
Received 18 June 2023; Received in revised form 20 August 2023; Accepted 27 August 2023
Available online 1 September 2023

in a variety of applications gaming, virtual reality, and healthcare to perform critical operational procedures. Using gestures to control a computer or a smart device, or enable a virtual character to mimic the movements of a user in a virtual reality environment would remove the requirement for an intermediate input device between humans and computers and users will be able to control machines with mere gestures [9]. Two main methods are being used for HGD, namely Static HGD and Dynamic HGD. Static HGD means the detection of still gestures whereas Dynamic HGD covers motion-based gestures. Over the years, researchers have developed a variety of ways for HGD including wearable technology-based, depth-based [10], and vision-based detection [11]. While wearable gloves can accelerate the process, it is important to note that the underlying technology can be costly and does not always ensure reliable outcomes. On the other hand, the availability of cheap but efficient sensors has played its part in the evolution of this field. Sensors like the Microsoft Kinect, Intel Realsense, and Sensz3D provide depth and color information, while the Leap Motion Sensor (LMC), employed in this study, captures short IR images and provides data regarding the skeletal information of the hand. The data can either is captured visually or it can be tabulated from the sensor data [12–14].

Another method, found in literature, is to use raw data directly and apply ML algorithms to identify gestures. This method has been discussed in [15] using the Deep Learning technique on more than 30000 features and in [16] using the k-nearest neighbors algorithm (KNN), Decision Trees, and support vector machines (SVM). The advantage of using raw data is that it is faster than the conventional algorithms used for image detection and with large enough data the efficiency could be improved greatly. However, not all features extracted from the sensor are useful and it is hard to go through each feature and find those that give the best results. In this regard, using feature selection techniques can narrow down the data set to the specific features which give the best results [17].

Feature selection (FS) is being used in a variety of applications for the process of reducing data sets without compromising on the efficiency of the system and in most cases even enhancing the performance as well by removing redundant features [9,18]. Dimensionality reduction using feature selection methods can be achieved in a variety of ways including selection based on statistical properties and correlations (Filter based methods) [19], using conventional techniques such as PCA [20] or LDA [21] and using Wrapper based methods. However, there is a new range of algorithms which, over the years have proven to be more effective in solving this problem for complex data sets, these are called metaheuristic algorithms. Metaheuristic algorithms have been widely used in feature selection to derive optimal feature subsets. These algorithms are renowned for their capability to discover the optimal solution within expansive feature spaces. They are commonly employed in optimization problems characterized by intricate search spaces and non-linear objective functions. Their effectiveness in navigating complex landscapes makes them valuable tools for tackling challenging optimization tasks. These are inspired by natural processes, such as evolution, hunting, movement, or foraging behavior of different organisms. Several bio-inspired meta-heuristic algorithms are commonly used in various optimization problems. Some of the most well-known bio-inspired meta-heuristic algorithms include Particle Swarm Optimization (PSO), Genetic Algorithm (GA), Artificial Bee Colony (ABC) Optimization, Firefly Algorithm (FA), Grey Wolf Optimization (GWO), and Whale Optimization Algorithm (WOA). Each of these algorithms is inspired by different biological phenomena, such as swarming behavior, evolution, migration, hunting, and foraging.

The LMC from Ultraleap is an optical hand-tracking device. It is primarily intended for use in interactive software applications for hand gesture and finger position recognition [22]. It records the motion of the user's hands and fingers to enable natural interaction with the digital content and has been widely used for gesture detection in various applications, including:

1. Human-Computer Interaction (HCI): the LMC can be used to provide a new form of interaction between humans and computers. With gesture recognition, users can control their computers and applications with hand and finger movements [23,24].
2. Virtual/Augmented/Mixed Reality (VR/AR/MR): the LMC can be integrated into VR/AR/MR systems to provide a more immersive and natural experience [25,26]. Users can use hand and finger gestures to control and interact with virtual environments.
3. Gaming: the LMC can find applications in gaming to provide a new interactive environment for users in the game. Players can control games with hand and finger gestures, providing a more immersive and intuitive gaming experience

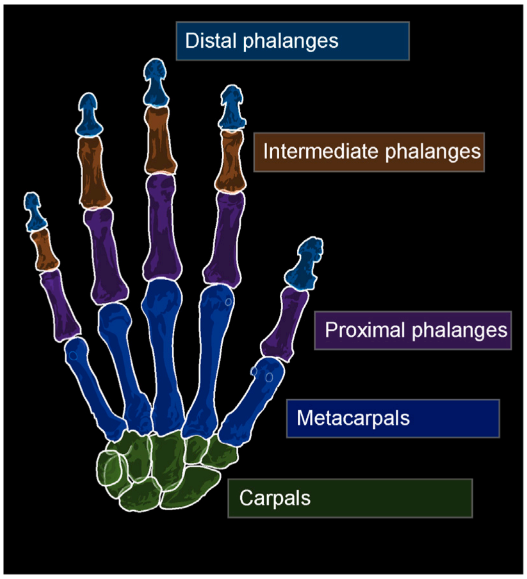
The LMC uses IR (Infrared) imaging to quickly ascertain the locations of preset objects within a constrained area. The controller follows hand movement in 3D interaction region that spans to more than 60 cm (24″) and has a nominal viewing field of 140° x 120°. The software provided by the manufacturer can capture 27 different hand features, such as bones and joints, and follow them even when they are hidden by other parts of the hands. Fig. 1(a) shows the reference hand model used by the LMC and a reference output image extracted using the visualizer provided by UltraLeap shown in Fig. 1(b).

The LMC employs two 640 x 240 pixel near IR cameras and three IR LEDs which are spaced equally. The LMC device structure is shown in Fig. 2. The sensor is integrated with the system using LeapMotion SDK provided by the manufacturer supported to use with Python. Data from LMC can be extracted both in image form and as spatial coordinates in tabular form. A total of 208 features are provided which consist of rotational and spatial features of the palm, fingers, and bones of each hand.

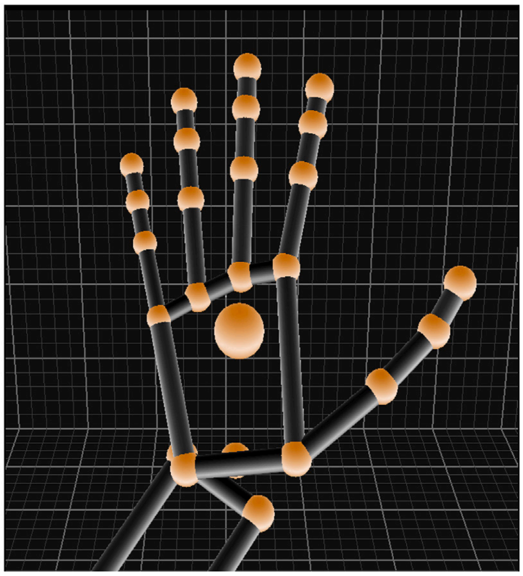
Datasets employing pictorial or moving frame information have been used extensively for HGD, however, in this paper we consider the possible use of spatial and spatiotemporal data for the purpose of gesture detection. As discussed earlier the LMC outputs 208 feature vectors and all features are not of equal quality, some may be redundant or carry no substantial information thereby degrading the performance of the model [28]. Employing intelligent algorithms for the process of feature reduction can greatly increase model efficiency and reduce computational expenses, thereby making the algorithm faster [29]. In order to determine the optimum subset for feature selection issues, such techniques as greedy, exhaustive, and random search have been used. The majority of algorithms, in literature, suffer problems like premature convergence, excessive complexity, and high computational expenses.

Over the last few decades, metaheuristic algorithms have emerged as promising approaches for addressing complex optimization problems. These algorithms provide effective solutions by leveraging intelligent search strategies and adaptive techniques, enabling them to navigate intricate search spaces and tackle non-linear objective functions. Their widespread application has shown great potential in solving diverse real-world problems and advancing optimization methodologies [30].

Metaheuristic algorithms are a new category of search and optimization algorithms that are inspired by different naturally occurring phenomena, a basic classification of these algorithms is shown in Fig. 3 [31]. Metaheuristic algorithms are more flexible and adaptable and can handle large amounts of data. Population-based search algorithms span wide search space efficiently and random search-based algorithms provide an efficient solution against local maxima problems. Moreover, bio-inspired algorithms are based on naturally occurring phenomena like hunting, foraging, and breeding behaviors of several animals [32]. PSO, GWO, and ABC are some of the most common nature-based bio-inspired metaheuristic algorithms. These algorithms use a population of organisms to scout a wide search space(exploration), then each population is evaluated against a cost function(exploitation) and finally, the populations converge on a solution [33,34]. During each iteration, randomness is introduced in



(a) Human Hand



(b) Image extracted from the LMC.

Fig. 1. Human Hand visualization using the LMC.

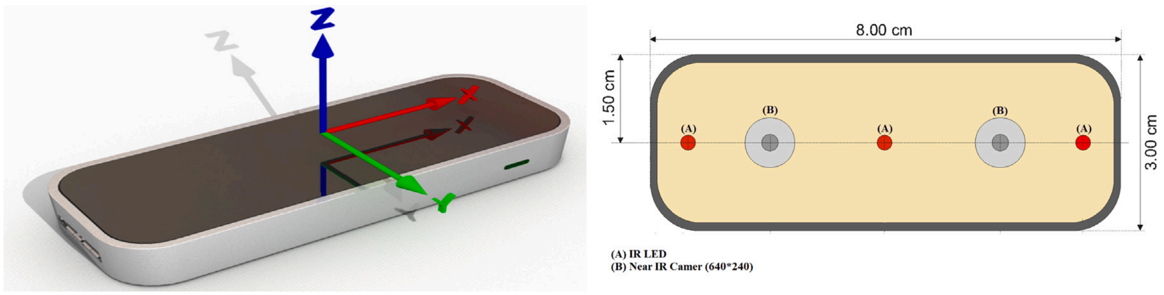
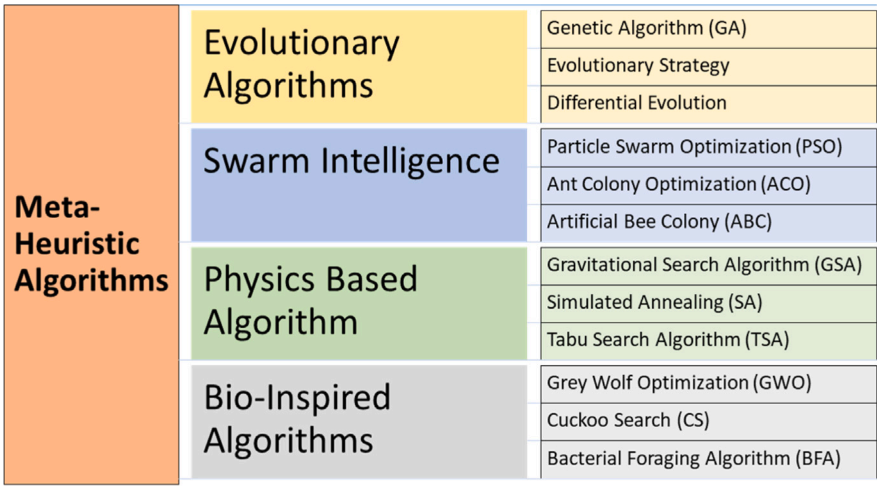


Fig. 2. LMC Schematic Diagram [27].



Meta-Heuristic Algorithms	Evolutionary Algorithms	Genetic Algorithm (GA)
		Evolutionary Strategy
		Differential Evolution
	Swarm Intelligence	Particle Swarm Optimization (PSO)
		Ant Colony Optimization (ACO)
		Artificial Bee Colony (ABC)
	Physics Based Algorithm	Gravitational Search Algorithm (GSA)
		Simulated Annealing (SA)
		Tabu Search Algorithm (TSA)
	Bio-Inspired Algorithms	Grey Wolf Optimization (GWO)
		Cuckoo Search (CS)
		Bacterial Foraging Algorithm (BFA)

Fig. 3. Brief Classification of Meta-Heuristic Algorithms.

the population so the solution(mutation) does not get stuck on local minima. The ability of these algorithms to deal with large dataset and complex real-world problems make them an ideal candidate for the purpose of dimensionality reduction.

In this paper, a dataset is developed containing eight static features using the LMC. The dataset is then pre-processed and passed through a metaheuristic algorithm. The first step is to use a hybrid metaheuristic algorithm for dimensionality reduction. Then the results are compared with conventional algorithms and it is empirically shown that the selected algorithm performs better than its conventional counterparts. The reduced dataset is then passed through a deep neural network(DNN)which is further trained by employing the same hybrid metaheuristic algorithm for the gesture classification purpose. At this stage, neural networks are trained for both reduced and complete

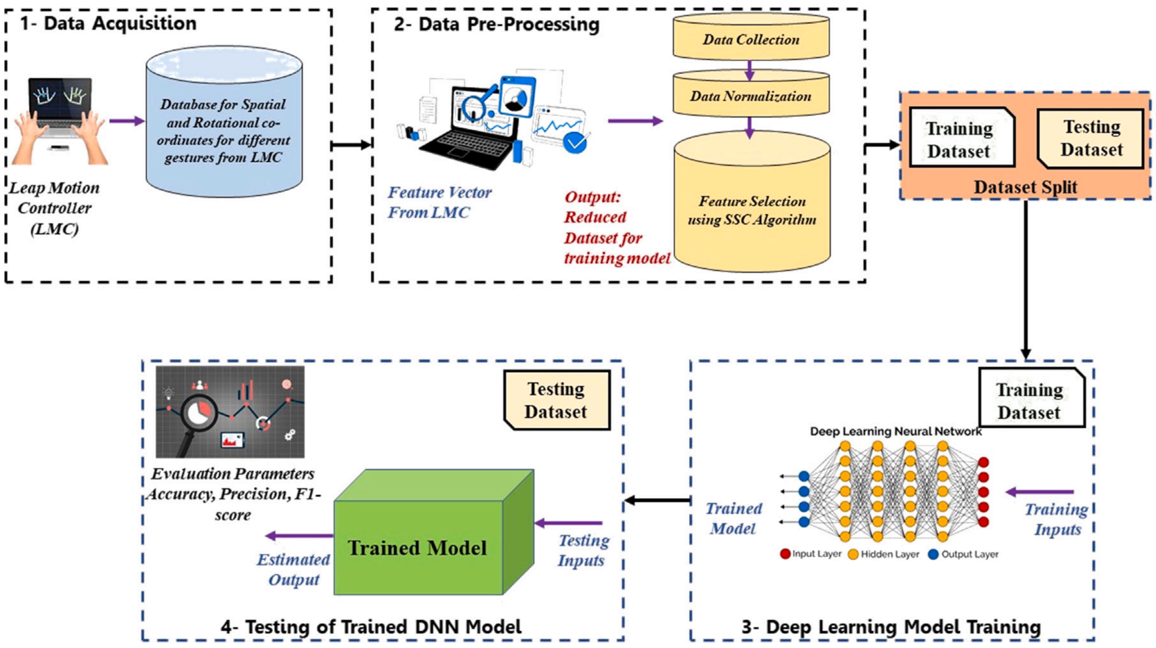

1- Data Acquisition
Database for Spatial and Rotational co-ordinates for different gestures from LMC
Leap Motion Controller (LMC)
2- Data Pre-Processing
Data Collection
Data Normalization
Feature Selection using SSC Algorithm
Feature Vector From LMC
Output: Reduced Dataset for training model
Training Dataset
Testing Dataset
Dataset Split
3- Deep Learning Model Training
Training Dataset
Deep Learning Neural Network
Trained Model
Training Inputs
Input Layer
Hidden Layer
Output Layer
4- Testing of Trained DNN Model
Testing Dataset
Evaluation Parameters Accuracy, Precision, F1-score
Trained Model
Estimated Output
Testing Inputs

Fig. 4. Proposed scheme of study for the proposed technique.

datasets. The results compiled show that reducing the dataset reduces the number of parameters to be trained and the complexity of the DNN without compromising the overall efficiency of the system. A detailed figure of the scheme developed during the course of this paper is shown in Fig. 4.

1.1. Contribution and paper organization

The contributions of this work are:

- Creation of a new hand gesture dataset that can potentially be used for future research in hand gesture recognition.
- Development of a novel metaheuristic algorithm (SSA) for feature extraction from a hand gesture dataset collected using the LMC.
- Comparison of the performance of the proposed metaheuristic algorithm with other state-of-the-art algorithms i.e. PSO, GWO and SCA for feature extraction from hand gesture data is evaluated.
- Development of a hybrid deep learning model (SSA-DNN) that integrates the proposed metaheuristic algorithm for feature extraction is formulated. The model can achieve better performance with selected and full features.
- The proposed methodology and experimental setup can potentially serve as a framework for other researchers interested in developing hand gesture recognition models.

The remainder of the paper is composed as follows. Section 2 discusses recent related work. In Section 3 details about the dataset generation and pre-processing is described. Section 4 highlights the details of the proposed technique for feature selection. Section 5 discusses the proposed technique for classification and Section 6 covers results and discussion of the experiment.

2. Related work

Metaheuristics have been the center for research in computing and optimization problems for the past few decades. GWO, ABC, PSO, and WOA are among some of the most widely used nature-inspired algorithms for solving complex real-world problems. PSO is a population-based algorithm derived from the collective movement of a flock of birds or a school of fish. It employs a swarm of particles to search for the best solution by adjusting their speed and position relative to their own experience and the experience of their neighbors [35]. GWO is a population-based optimization algorithm inspired by the hunting behavior of grey wolves. It mimics a wolf pack's leadership hierarchy and hunting mechanism to search for optimal solutions [36]. ABC is a population-based optimization algorithm inspired by the foraging behavior of honey bees. It uses a population of artificial bees to cover the search space and determine the best global solution [37]. WOA is a meta-heuristic algorithm inspired by the hunting behavior of humpback whales. It mimics the bubble-net hunting technique used by humpback whales to find optimal solutions [38]. More recent hybrid algorithms have also been developed that use multiple meta-heuristic techniques or a combination of meta-heuristic and optimization algorithms to solve the problems more effectively [39]. Azzougui et al. [40] used GWO to find the optimal placement of phasor measurement units in a wide area network and applied the technique on the Algerian 63 bus system.

Researchers have developed a significant interest in the field of human–computer interaction (HCI) in recent years, and one of the key areas within this field is gesture detection. Researchers have studied multiple gesture detection methods including wearable technology, vision-based and depth-based detection. In [41], the use of accelerometers and gyros have been discussed whereas [42,43] discuss the use of multi-touch sensors to capture hand and finger movements. Depth sensors, a fairly new technology as compared to other HGD techniques [44, 45], show image-based feature extraction and gesture detection using LMC. Yao et al. [46] discussed the application of Microsoft's Kinect for the purpose of HGR. Zeng et al. [47] proposed a new technique for HGR using LMC via deterministic learning. The method extracts features from LMC and models hand motion dynamics using constant radial basis function neural networks. Using different cross-validation styles, the method achieves good accuracy for the classification of English alphabets and Arabic numerals. Lu et al. [48] presented a method for dynamic HGR using LMC and a novel feature vector. The method uses a Hidden Conditional Neural Field classifier to determine gestures based on their depth information. Xu et al. [49] presented a HGR system using feature extraction and NN. The system consisted of four stages: image acquisition, data pre-processing, feature extraction, and classification. The system was able to recognize 10 hand gestures from the American Sign Language (ASL) alphabet. The study compared different feature extraction methods and different neural network architectures and evaluated the performance of the system using accuracy, precision,

Table 1
Data extracted from the LMC.

Gesture	Hand	PalmX	H_Roll	H_Yaw	Arm_DirX	Wrst_Pos_Z	EL_Pos_X	EL_Pos_Y
1	R	10.92	−8.24	−0.31	−0.27	123.45	93.59	224.07
2	R	132.8	145.16	44.97	0.86	99.03	−157.31	43.55
3	R	20.00	−9.68	−58.91	−0.92	57.37	338.81	152.56
4	L	−4.63	−9.20	4.75	0.015	134.67	−20.77	319.30
5	R	2.24	−0.97	−5.88	−0.58	97.34	182.98	171.45
6	R	−1.27	8.35	−17.75	0.67	108.03	−166.58	30.61
7	L	7.21	66.96	−11.15	−0.44	96.33	141.36	250.18
8	R	−33.41	81.07	4.21	0.16	68.42	−87.66	233.00

recall, and F-measure metrics. In [15], the authors presented a HGR system using LMC and DNN. The system reconstructed the palm model and extracted the feature data from LMC. The system trains the DNN model with the normalized data to achieve good gesture recognition accuracy. For the dataset, 30 features were captured from the LMC mainly focusing on the spatial positioning of the palm and bones. The paper focused on the classification of three gestures by employing static feature selection and classification using a DNN model.

3. Experimental setup

For this experiment the LMC was mounted on a jig to keep the position static. The data was collected under controlled lighting conditions throughout the experiment as IR sensors are sensitive to light. During the course of the experiment, data was collected with varying distances ranging from 100 mm to 300 mm to add variation in scale. The LMC was integrated with the system using LEAPSDK v2.0 and python 2.7 on a Hp core i3-10th generation Intel micro-processor and no GPU. The data set was collected and stored as .csv file. This data was then imported to Python 3 and pre-processed using a variance-based method for initial scrutiny of the feature set. After the first pre-processing the data was normalized and passed through the feature selection and classification algorithms.

3.1. Dataset generation

Data for eight static hand gestures was recorded for both left and right hands with 16 participants. A mixed variety of participants was included from different age groups and genders to introduce variation in scale. Data was recorded at different heights for both Left and Right hands and some gestures were skewed intentionally to introduce variation in orientation to cater to different real-world scenarios. Previous datasets available online use pictorial representation however, for the purpose of this paper tabular data is collected to train the proposed model. The LMC gives 210 feature points for each sample point. A sample of entry for each class is shown in Table 1. Out of these only 19 features were dropped due to negligible variance. The dataset contains positional and rotational data for the wrist and palm. The sensor also gives the position of four bones(meta-carpel, proximal, intermediate, and distal) of each finger. A total of 264 sample points were collected for five labels. Some of the feature points are shown in Fig. 5. To further improve model performance data was normalized to create a uniform distribution. It was observed that the model gave good and consistent results when using normalized data. Table 1 displays several data points. Fig. 6 shows the correlation of data points plotted between different features against gesture classes. It is evident that simply choosing random features will not help in designing a good model for gesture detection as the data points are not distributed distinctly.

3.2. Dataset pre-processing

Data pre-processing in this case mainly comprises data normalization. The minima and maxima of each feature are extracted and data is normalized as per (1). The raw data from the LMC includes information about the position and rotational features of hands, fingers, and bones. This data is pre-processed and transformed into a set of features that represent the gesture. Using a standard deviation depended algorithm features with less information are removed. Data is normalized during the pre-processing phase as this can improve model performance, prevent data biases, and increase its convergence speed.

$$\xi'_a = \xi_a - \frac{min_A}{max_A - min_A} \tag{1}$$

where ξ_a shows the current value, min_A and max_A are the minimum and maximum value of Ath column.

4. Proposed technique for feature selection

The scientific community is growing keen on feature selection significantly, the best traits from a preliminary dataset are selected to enhance the quality of features in numerous ML tasks, including classification, regression, clustering, and time-series activity prediction [50,51]. Dimensionality reduction techniques are a key area of research in data mining and machine learning. They are intended to select the best subset of relevant features from an original dataset using specific evaluation criteria, which results in better learning performance, better model interpretability, lower computational costs, and greater learning accuracy. The objective function is typically defined in terms of the accuracy, efficiency, or other performance measures of a machine learning model that is trained on the selected features.

In a context where time and computing resources are restricted, achieving optimal solutions for complex optimization problems becomes challenging. This is where metaheuristic algorithms come into action. Metaheuristic algorithms are high-level problem solvers designed to generate satisfactory solutions for difficult optimization problems that are hard to solve optimally. These methods excel in situations where data may be incomplete or imperfect. Instead of relying on exhaustive search or exact algorithms, they employ heuristics and iterative processes to explore the solution space and converge towards promising solutions. By leveraging their flexibility and adaptability, Metaheuristic algorithms offer a practical approach to finding good solutions within reasonable time constraints. They are particularly useful in situations where traditional optimization techniques may struggle due to the complexity and computational demands of the problem at hand. Feature selection is one of the most tedious optimization problems, several metaheuristic techniques such as PSO, WOA, GWO and BChOA, have been employed to find the optimal solution with accuracy and efficiency [52]. In [53], the authors have used Binary Chimp Optimization Algorithm to reduce the feature map of the biomedical dataset for classification. In [54], Dragonfly Algorithm has been used for the dimensionality reduction of a Drug Bank dataset. This study utilizes a new class of hybrid metaheuristic algorithms that has been developed by combining other basic metaheuristic algorithms and has been proven to be more effective as compared to the parent algorithms. This study discusses the use of a Spotted hyena-based Sine-cosine Chimp optimization (SSC) algorithm for the purpose of feature selection.

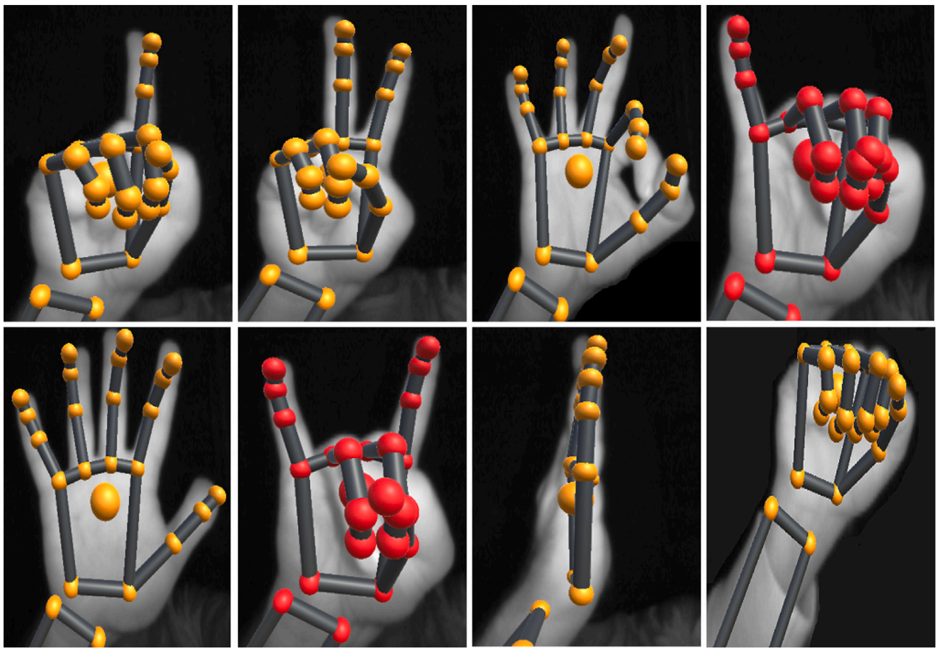
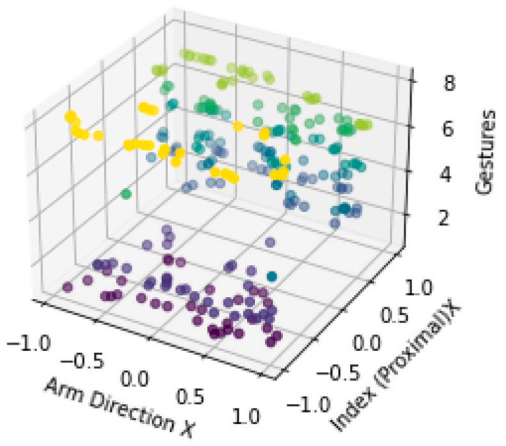
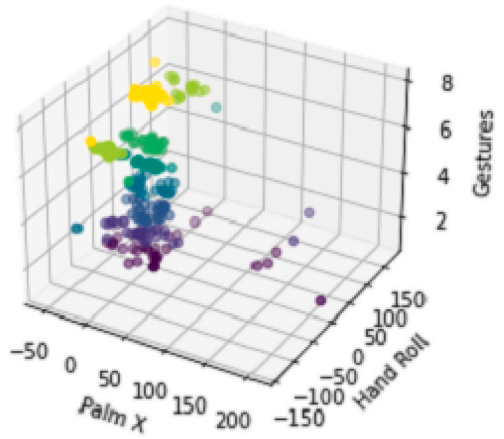


Fig. 5. Gestures used for detection.



(a) Arm Direction (x) vs Pinky Distal (z) vs Gesture Classes (y).



(b) Palm Position(x) vs Hand roll (z) vs Gesture Classes (y).

Fig. 6. Correlation sample between different features on the x and y axis vs. Gesture classes on the z-axis.

4.1. SSC metaheuristic algorithm

As discussed in Section 1.1, the dataset developed for gesture detection contains 191 features. Solving a classification problem for such a dataset using conventional methods will not yield useful results. The hybrid algorithm, SSC [55], used is based on the Sine-Cosine Algorithm (SCA) [56], Spotted Hyena Optimizer (SHO) [57] and Chimp Optimization Algorithm (ChOA) [58]. The algorithm is inspired by the sexual attraction of chimps and the exploration of spotted hyenas. The sine-cosine algorithm has been used to introduce randomness in order to solve the local minima problem. The initialization process of ChOA along with the attacking strategy of SHO and update rule as per SCA has shown good results as compared to the individual algorithms. The use of SCA helps to avoid local minima while the ChOA widens the search space for better global minima tracking.

4.1.1. Sine-cosine algorithm

SCA was originally presented by Mirjalili in [59], the algorithm utilizes the mathematical model of sine-cosine functions to fluctuate the population towards or away from the optimal solution. By introducing the randomness based on these oscillatory functions this system has shown promising results to tackle the local optima problem. The solution update mechanism for the algorithm is depicted as:

$$\zeta_i^{k+1} = \zeta_i^k + (R_1 * sin(R_2) * |R_3.\zeta_i^k - \xi_i^t|) \tag{2}$$

$$\zeta_i^{k+1} = \zeta_i^k + (R_1 * cos(R_2) * |R_3.\zeta_i^k - \xi_i^t|) \tag{3}$$

where ζ_i^k is the current position in ith dimension at kth iteration and R_1, R_2, R_3 are random numbers and ξ_i^t is the solution for ith iteration and kth population. A graphical representation of SCA is given in Fig. 7.

4.1.2. Spotted hyena algorithm

The spotted hyena algorithm is based on social activity and the relation between hyenas. It mainly focuses on the attacking, encircling, hunting, and searching behavior of hyenas. SHO mimics the activity of spotted hyenas to achieve an optimum solution. Encircling can be depicted by the following two equations.

$$\overrightarrow{\zeta_h} = |A.\overrightarrow{\xi_p}(x) - \vec{\xi}(x)| \tag{4}$$

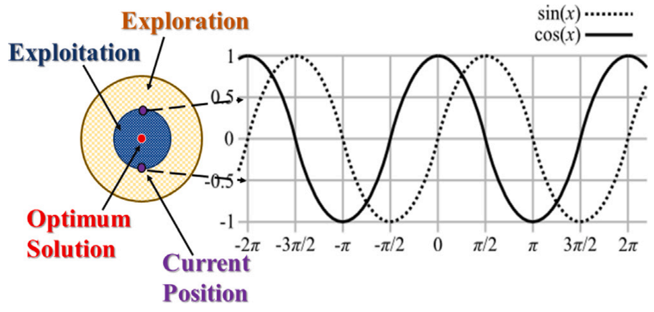


Fig. 7. Metaheuristic: Sine-Cosine Optimization Algorithm.

$$\vec{\xi}(x+1) = \vec{\xi_p}(x) - B.\vec{\zeta_h} \tag{5}$$

where ξ_p show the optimum feature vector (FV), $\xi(x)$ is the current FV and ζ_h shows the distance that the corresponding hyena would have to travel to catch its prey. The coefficients A and B are calculated using the following equations

$$A = 2.d_1 \tag{6}$$

$$B = 2.h.d_2 - h \tag{7}$$

$$h = 5 - \frac{k^{th} * 5}{max_k} \tag{8}$$

here, the value kth is the value of current iteration and max_k is the maximum number of iterations. The vector h is reduced from 5 to 0, as shown in (8), and d_1 and d_2 are random numbers within the range of $[0,1]$. Following equations map the hunting regions of spotted hyenas:

$$\vec{\zeta_h} = |A * \vec{\xi_h} - \vec{\xi_k}| \tag{9}$$

$$\vec{\xi_k} = \vec{\xi_h} - B * \vec{\zeta_h} \tag{10}$$

$$\vec{O_h} = \vec{\zeta_k} + \vec{\zeta_{k+1}} + \cdots + \vec{\zeta_{k+N}} \tag{11}$$

where $\vec{\xi_k}$ shows the search space of the corresponding hyenas and $\vec{O_h}$ shows the cluster of optimal solutions and N is the number of iterations which can be calculated as.

$$N = count_n(\vec{\zeta_h}, \vec{\zeta_{h+1}}, \vec{\zeta_{h+2}}, \ldots, \vec{\zeta_{h+M}}) \tag{12}$$

$$\vec{\zeta}(x+1) = \frac{\vec{O_h}}{N} \tag{13}$$

where, n is the number of solutions that are in a region close to the optimal solution. The vector $\vec{M}$ shows a randomly initialized vector between $[0.5,1]$. The best solution $\vec{\zeta}(x+1)$ helps in updating the remaining solutions at the end of each iteration. The exploration of the hyenas is ensured using the random coefficients A and B where the exploration starts when $-1 < B < 1$ and A ranges between $[0,5]$ and serves as a weight for the corresponding hyena. The optimization process begins with the initialization of a random population, each hyena marks its territory and with each iteration, the coefficients h are decreased linearly. The best agents are then fetched after each iteration.

4.1.3. Chimp optimization algorithm

The Chimp Optimizer algorithm is a metaheuristic optimization algorithm inspired by the behavior of chimpanzees in their natural habitat. It mimics the social behavior of chimpanzees in a group, where each member of the group interacts with its peers to find the best solutions to a problem, as illustrated in Fig. 8. This technique was first developed by Khishe et al. in [58].

The mathematical model of ChoA is defined as follows

$$\beta_r = |\gamma.x_{prey}(i) - m.x_{chimp}(i)| \tag{14}$$

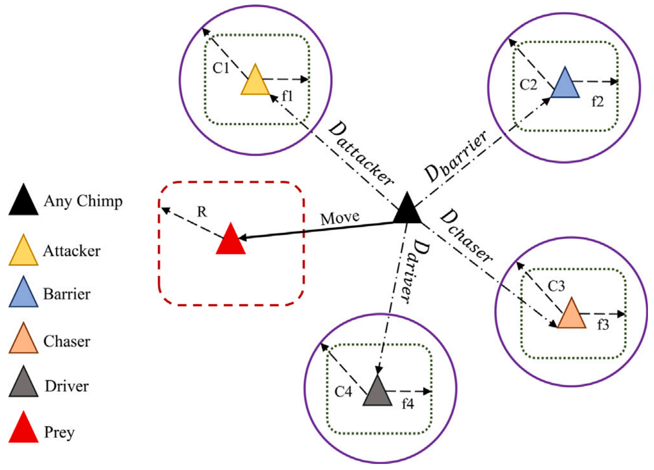


Fig. 8. Working mechanism of ChoA Algorithm.

$$x_{chimp}(i+1) = x_{prey}(i) - \alpha.\beta \tag{15}$$

where i is the number of current iterations, x_{prey} is the position of prey and x_{chimp} is the chimp position. The vectors α, m, and γ can be calculated as:

$$\alpha = 2.f.r_1 - f \tag{16}$$

$$\gamma = 2.r_2 \tag{17}$$

During the iterative process the coefficient f is reduced form 2.5 to 0, r_1. and r_2 are random values in the range [0,1]. The attacking model of chimps can be described by the following equations.

$$\delta_{attacker} = |c_1.\alpha_{attacker} - m_1.x| \tag{18}$$

$$\delta_{barrier} = |c_2.\alpha_{barrier} - m_2.x| \tag{19}$$

$$\delta_{chaser} = |c_3.\alpha_{chaser} - m_3.x| \tag{20}$$

$$\delta_{diver} = |c_4.\alpha_{diver} - m_4.x| \tag{21}$$

The initial chimps position is determined by the value of random vectors. The chimp's next position will be between its current position and prey when the value is between the range [−1,1]. The update strategy follows the following set of equations.

$$x_n = \alpha_{attacker} - a_n.\delta_{attacker} \tag{22}$$

$$x_2 = \alpha_{barrier} - a_2.\delta_{barrier} \tag{23}$$

$$x_3 = \alpha_{chaser} - a_3.\delta_{chaser} \tag{24}$$

$$x_4 = \alpha_{diver} - a_4.\delta_{diver} \tag{25}$$

$$x_{t_i+1} = \frac{x_1 + x_2 + x_3 + x_4}{4} \tag{26}$$

The following equation is applied to update each chimps position where the value m is the chaotic value obtained randomly for each chimp and the distribution of m can be chosen based on the optimization problem.

$$\alpha_{chimp}(n_i+1) = \begin{cases} \alpha_{prey}(n_i) - x, \Delta, & \text{if } \phi < 0.5 \\ m, & \text{if } \phi > 0.5 \end{cases} \tag{27}$$

4.2. Mathematical model of the SSC hybrid algorithm

Due to the complex nature of the gesture detection optimization problem, the hybrid algorithm is selected for this study. The SSC algorithm combines the attacking strategy of SHO along with the ChoA and optimizes the update rule with the SCA. This new strategy allows the algorithm to cover more search space and the randomness, introduced by the SCA, tackles the local minima problem thus making the algorithm more efficient as compared to the individual algorithms. The complete algorithm can be mathematically expressed as

$$x_1 = a_{attacker} - cos(r_2) * a_1.d_{attacker} \tag{28}$$

$$x_2 = a_{barrier} - sin(r_2) * a_2.d_{barrier} \tag{29}$$

$$x_3 = a_{chaser} - cos(r_2) * a_3.d_{chaser} \tag{30}$$

$$x_4 = a_{diver} - sin(r_2) * a_4.d_{diver} \tag{31}$$

Here $r_2 = (2\pi) * rand$ and lies in the range $[0, 2\pi]$. The following equation is used to update the location of chimps during searching.

$$x_i = \frac{x_1 + x_2 + x_3 + x_4}{4} \tag{32}$$

The variable $x_i (i = 1, 2, 3 \dots n)$ shows the chimp population.

4.3. SSC for feature selection

The Feature Selection using the SSC algorithm operates as follows.

1. Initialize population of x_i for $(i = 1, 2 \dots, N)$
2. Evaluate each individual using the fitness function.
3. Update the feature subsets of the individuals based on the Eqs. (28)–(31)
4. Repeat steps 2 and 3 until a stopping criterion is met.
5. Select the feature subset with the best fitness value as the final result.

A set of random features is initially selected and evaluated using a fitness function given by Eq. (33). The fitness function J is defined as:

$$J(S) = \frac{1}{m} * \Sigma_{i=1}^{m} [y(i) \neq g(x(i), S)] \tag{33}$$

where S is the feature subset, $g(x_i, S)$ is the prediction of the machine learning model for instances $x(i)$ using the feature subset S, $y(i)$ is the total number of classes, and $y_i \neq g(x_i, S)$ is an indicator function that is equal to 1 if the prediction is incorrect and 0 otherwise. The overall flow of the procedure for feature selection is illustrated in Fig. 9. Multiple population sizes were tested and it was found that a population size of $N = 20$ gives the best results in terms of efficiency and time. The experiment was run for 60 iterations for each 20/80 train-test split. Overall, 30 iterations were run for each algorithm i.e. SSC, GWO, PSO, and SCA, to compare the results.

5. Proposed technique for classification

5.1. Deep learning model (DNN)

A three-layer DNN model is employed in this study which consists of an input layer, three hidden layers, and an output layer. The entire DNN is densely connected and the output of each neuron in the hidden layers is calculated using the weighted sum of the inputs from the previous layer, followed by the application of a non-linear activation function. The output of the final layer is the predicted output of the model.

Let x be the input to the network, $w_{ij}^{(k)}$ be the weight between the ith neuron in the $(k-1)$th layer and the jth neuron in the kth layer, $b_j^{(k)}$ be the bias of the jth neuron in the kth layer, $z_j^{(k)}$ be the weighted sum of inputs to the jth neuron in the kth layer, and $a_j^{(k)}$ be the output of the jth neuron in the kth layer. Then, the equations for computing the output of the three-layer perceptron are:

$$z_j^{(1)} = \sum_i w_{ij}^{(1)} x_i + b_j^{(1)} \tag{34}$$

$$a_j^{(1)} = f(z_j^{(1)}) \tag{35}$$

$$z_j^{(2)} = \sum_i w_{ij}^{(2)} a_i^{(1)} + b_j^{(2)} \tag{36}$$

$$a_j^{(2)} = f(z_j^{(2)}) \tag{37}$$

$$z_j^{(3)} = \sum_i w_{ij}^{(3)} a_i^{(2)} + b_j^{(3)} \tag{38}$$

$$a_j^{(3)} = f(z_j^{(3)}) \tag{39}$$

$$y = f(z_j^{(4)}) \tag{40}$$

where f is the activation function, and y is the output of the network.

In the above equations, $z_j^{(1)}$ represents the weighted sum of the inputs in the first hidden layer, $a_j^{(1)}$ represents the output of the first hidden layer, $z_j^{(2)}$ represents the weighted sum of the inputs in the second hidden layer, $a_j^{(2)}$ represents the output of the second hidden layer, $z_j^{(3)}$ represents the weighted sum of the inputs in the third hidden layer, $a_j^{(3)}$ represents the output of the third hidden layer, and $z_j^{(4)}$ represents the weighted sum of the inputs in the output layer.

The input data is forward propagated through the DNN network to generate error which is used to optimize the weights and biases of the network using the SSC algorithm. As shown in Fig. 10, the back-propagation algorithm is replaced with the proposed SSC technique.

5.2. Hyper-parameters of DNN

Hyper-parameters are parameters in a NN that are set prior to training and remain fixed during training. In a DNN, the hyper-parameters include the number of neurons in each hidden layer, the learning rate, the activation function, and the regularization strength. Tuning these hyper-parameters is crucial for achieving optimal performance of the neural network. For example, increasing the number of neurons in a layer may improve the network's ability to model the complex combinations in the data, but may also increase the risk of over-fitting. Similarly, choosing an appropriate activation function is crucial as it can impact the performance of the network. Therefore, careful tuning of hyper-parameters is necessary to optimize the performance of a DNN.

In a DNN with three hidden layers, the weights and biases are important parameters that are learned during training to optimize the network's performance. The weights represent the strength of the connections between neurons in adjacent layers, while the biases represent the threshold at which a neuron is activated.

Effective training of the weights and biases is essential for achieving optimal performance of the neural network. If the weights and biases are not properly trained, the network may suffer from over-fitting or under-fitting, which can lead to poor performance on unseen data. Overfitting happens when a neural network is overly complex and perfectly fits the training data, resulting in poor performance on new data. Under-fitting, on the other hand, occurs when the network is too simple and fails to capture the underlying patterns in the data.

Proper regularization techniques such as L1/L2 regularization, dropout, and early stopping are essential to combat over-fitting and under-fitting. They add penalties, promote model simplicity, prevent over-reliance, and halt training when necessary, ensuring optimal performance and generalization. Dropout randomly drops out some neurons during training, which helps to prevent over-fitting by forcing the network to learn more robust representations. Early stopping halts the training process before the NN overfits the data by tracking the outcome of the network on validation data and when there is no further improvement in the results.

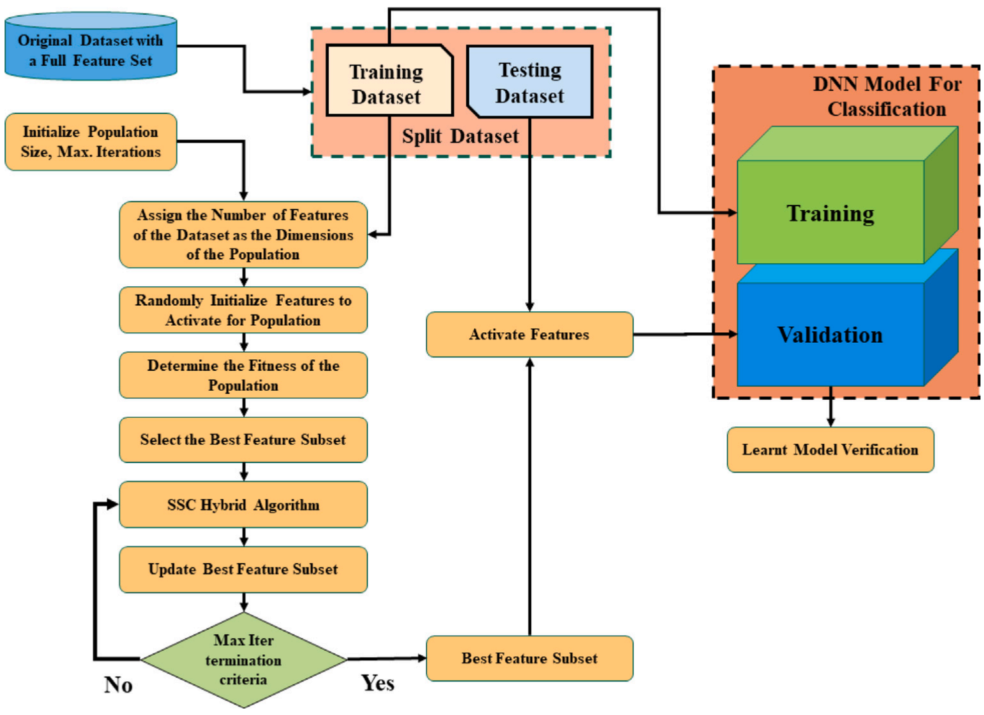

Original Dataset with a Full Feature Set
Training Dataset
Testing Dataset
Split Dataset
Initialize Population Size, Max. Iterations
Assign the Number of Features of the Dataset as the Dimensions of the Population
Randomly Initialize Features to Activate for Population
Determine the Fitness of the Population
Select the Best Feature Subset
SSC Hybrid Algorithm
Update Best Feature Subset
Max Iter termination criteria
No
Yes
Best Feature Subset
Activate Features
DNN Model For Classification
Training
Validation
Learnt Model Verification

Fig. 9. Flow diagram of SSC Hybrid Meta-heuristic Algorithm.

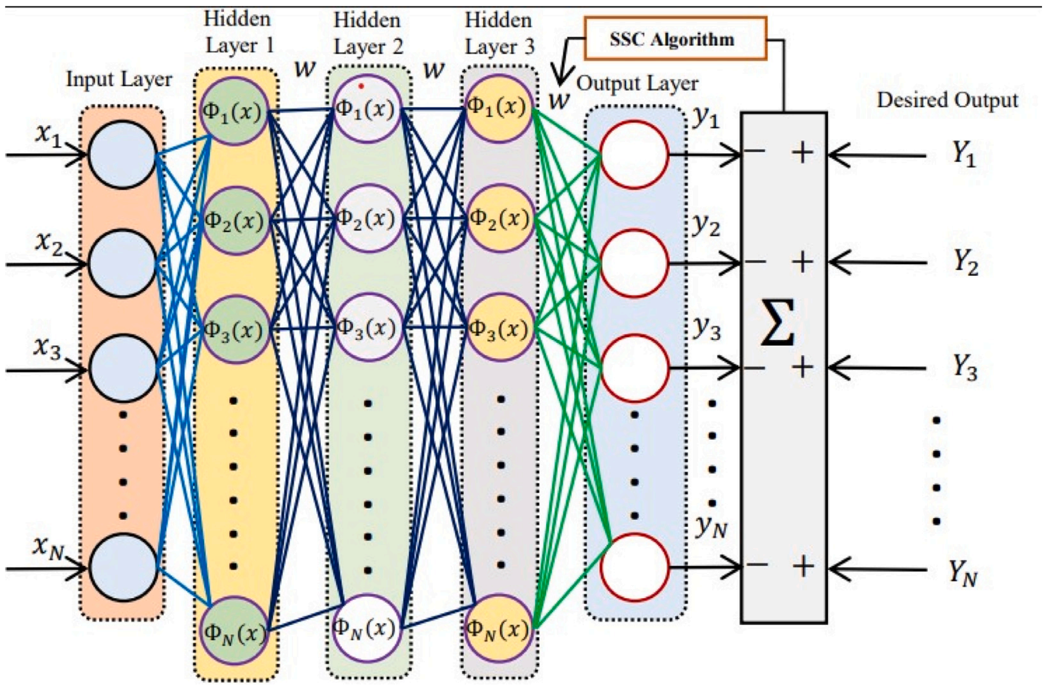


Fig. 10. SSC based Three Layer DNN structure for Classification.

5.3. SSC-based DNN model

Metaheuristic algorithms have been used in recent years for the training of weights and biases in neural networks. Compared to traditional optimization algorithms such as stochastic gradient descent, metaheuristic algorithms offer several advantages. Firstly, they are less sensitive to initialization and are able to escape local optima more easily, allowing them to explore a larger search space. Secondly, they can handle non-convex and non-smooth optimization problems, which are common in deep learning. Moreover, they can be used in parallel and distributed computing environments, enabling the optimization of large-scale NNs. Fig. 11 represents the flowchart for the program used to determine the trainable parameters of the DNN.

Presently, there are no specific research papers that have utilized the SSC for the training of weights and biases in NNs. However, given the promising performance of SSC and its variants in optimization problems, it is possible that the SSC could be used for optimizing the weights and biases of neural networks. The proposed algorithm offers

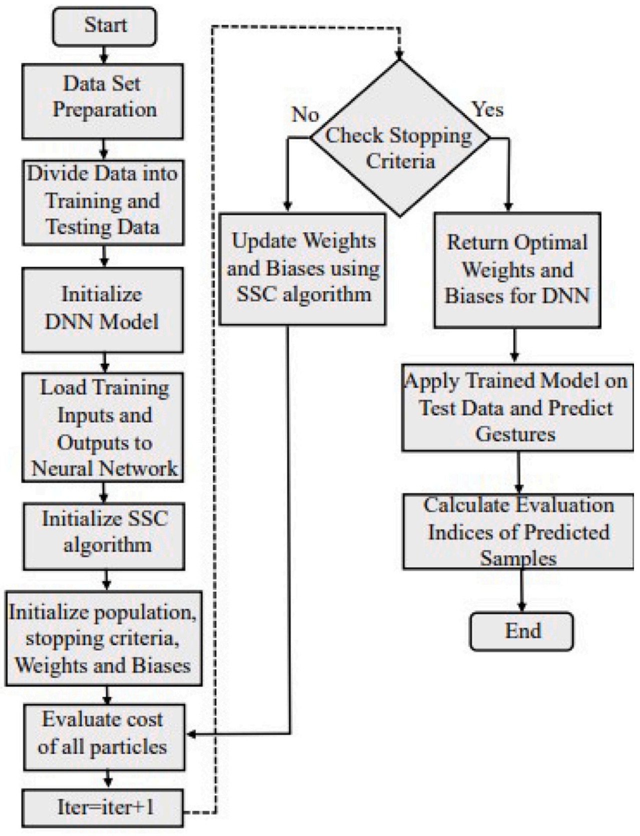

Start
Data Set Preparation
Divide Data into Training and Testing Data
Initialize DNN Model
Load Training Inputs and Outputs to Neural Network
Initialize SSC algorithm
Initialize population, stopping criteria, Weights and Biases
Evaluate cost of all particles
Iter=iter+1
Check Stopping Criteria
No
Yes
Update Weights and Biases using SSC algorithm
Return Optimal Weights and Biases for DNN
Apply Trained Model on Test Data and Predict Gestures
Calculate Evaluation Indices of Predicted Samples
End

Fig. 11. Flow chart of SSC-based Tuning of Weights and Biases of DNN.

improved performance compared to traditional optimization algorithms for neural network training. Therefore in this work, SSC is used for effective tuning of weights and biases of the DNN.

6. Results

Results for this paper are recorded for both feature selection and classification algorithms. The feature selection method is compared with conventional metaheuristic algorithms over multiple iterations and different metrics are recorded for each algorithm. Section 6.1 discusses the results of feature selection. In addition, The selected features are then passed through a DNN trained with different metaheuristic algorithms, and the results of the DNN classification are discussed in 6.2.

6.1. Evaluation of feature selection

During the experimentation, 30 iterations were performed. The train-test split ratio was set at 0.2 for test set. In each iteration, different train-test data was used. Accuracy, Precision, F1-score, execution time, and total reduced features were recorded for each iteration. Final data was tabulated and average values were tabulated to check the final results. Each algorithm is trained using Random Forest (RF) and SVM optimizer and all metrics are compared for PSO, GWO, SCA, and SSC algorithms. Fig. 12 shows a comparison of average scores for accuracy, precision, F1-score, and no. of features extracted using the feature selection algorithm. Fig. 13 shows the average train time for each technique over 30 different iterations. It is evident that the proposed SSC algorithm has the highest scores among the metaheuristic algorithms whilst recording the lowest average among the features selected. Fig. 14 shows the best results achieved during the 30 iterations, showing a similar pattern, with SSC achieving the highest accuracy, precision, and F1-score and lowest no. of features among the selected algorithms. These bar charts show a comparison of the selected techniques with the proposed algorithm for RF and SVM optimizers.

Fig. 15 shows the correlation among the selected features extracted using the SSC algorithm. Here we can see that some features possess strong correlations with each other however, the majority of the features do not have strong co-linearity. This shows that each feature is independent and does not cause a drift in the subsequent feature to drift. This weak correlation is often helpful in avoiding over-fitting when training DNN as the features remain distinct and introduce strong generalization.

Table 2
Classification results evaluation with full features.

Technique	Accuracy (%)	F1 score (%)	Precision (%)	Recall (%)
SSC-DNN	96.15	94.72	96.11	94.44
AOA-DNN	92.31	92.76	91.87	94.44
GWO-DNN	90.38	89.93	90.58	90.36

Table 3
Classification results evaluation with selected features.

Technique	Accuracy (%)	F1 score (%)	Precision (%)	Recall (%)
SSC-DNN	98.08	96.76	98.61	95.83
AOA-DNN	96.15	96.56	96.53	97.05
GWO-DNN	94.23	94.08	94.44	94.27

6.2. Evaluation of SSC-DNN for classification

The features selected by the SSC algorithm in Section 4.3 are passed through a DNN trained with the SSC algorithm. The results from the reduced dataset are compared with DNN trained with full features. Figs. 16 and 17 show the confusion matrix for selected and full features. It is evident from the recorded data that the efficiency of DNN trained with selected features is comparable with the results from full features. Table 2 shows different metrics recorded during the experiment with the DNN trained using PSO, GWO, and SSC for full features and Table 3 shows the recorded data for selected features. It is evident from the data that the SSC outperforms other techniques for the purpose of feature selection and classification from the data collected from the LMC.

6.3. Comparative study

Table 4 shows a comprehensive comparative analysis of different techniques for gesture detection using feature selection and metaheuristic algorithms highlighting their respective strengths and weaknesses. The discussed techniques include traditional feature selection methods such as wrapper, filter, PCA and embedded approaches, as well as metaheuristic algorithms such as GA, PSO, and the SSC algorithm proposed in this study. The comparative analysis provides valuable insights into the performance and applicability of these techniques for accurate and efficient gesture detection.

Notably, our proposed model, combining Meta-heuristic SSC-based feature selection with DNN training, achieves exceptional accuracy of 0.98. This outcome highlights the model's efficacy in optimizing feature selection to enhance gesture detection accuracy, particularly valuable in resource-constrained scenarios. The integration of SSC across both feature selection and DNN training showcases the model's synergistic approach, ultimately outperforming other techniques and demonstrating its potential as a superior solution in this domain.

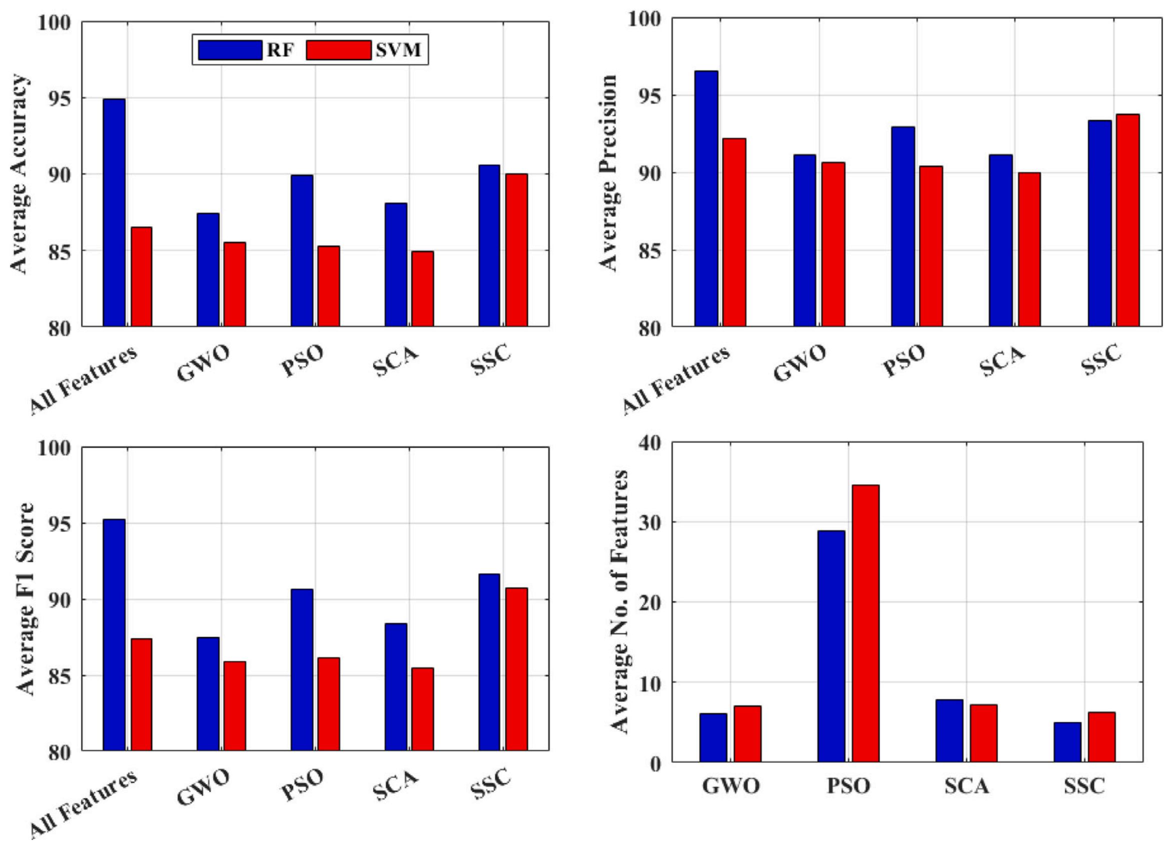


Fig. 12. Feature Selection Results: Average Accuracy, Precision, F1-Score, No. of Features.

Table 4
Comparative analysis of feature selection and meta-heuristic algorithms for gesture detection.

Title	Data-set	Technique used	Accuracy
[60]	Indian Sign Language(Pictorial 6 static colored images)	Feature selection + NN for classification (trained using hybrid meta-heuristic deer hunting + gwo)	0.97
[61]	Image Data-set from Kaggle (36 columns)	PCA + BPSO for NN training	0.98
[62]	Spatial Data-set (Arabic Sign Language Numerals 1-9)	Radial basis function	0.94
[63]	Image Data-set (Arabic Sign Language, 14 images)	KNN/SVM + Ada Boost	0.91
[64]	Ten Static Images Collected Using LMC	FS(using GA) + KNN/RF/NB	0.74
[65]	ASL (26 Letters Static)	SVM + DNN	0.94
[64]	ISL (26 Letters)	NB	0.95
[49]	Static Nine gestures	FS(static; based on Orientation) + SVM classification	0.91
[66]	Six Static images	HOG + SVM (multi-class)	0.92
[15]	Three static gestures (Rock, paper, scissors)	FS (static 11 points) + DNN Classification	0.98
Our model	Eight static gestures	Meta-heuristic (SSC) FS + DNN Trained with SSC	0.98

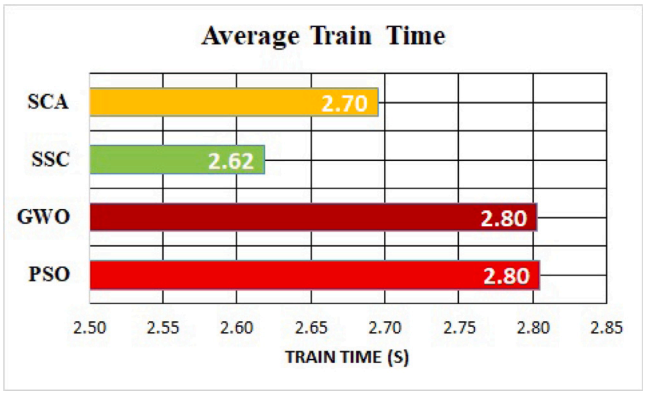


Fig. 13. Feature Selection Results: Average Train times over 20 iterations.

7. Conclusion

In this research paper, we presented the development and evaluation of a novel technique for feature selection using a hybrid meta-heuristic algorithm called SSC (Spotted hyena-based Chimp optimization algorithm). Our algorithm aimed to address the challenges of dimensional reduction in gesture recognition using the Leap Motion Controller (LMC) device. Through extensive experimentation and comparison with existing algorithms, we have demonstrated the superior performance of SSC across multiple metrics.

The results obtained from our experiments clearly indicate that SSC outperforms other algorithms, namely GWO, PSO, and SCA in terms of accuracy, precision, F1-score, time to select features, and the number of features. This significant improvement in performance showcases the effectiveness of SSC as a robust solution for dimensional reduction in gesture recognition tasks. The proposed algorithm not only achieved superior results but also exhibited consistency and stability across various datasets.

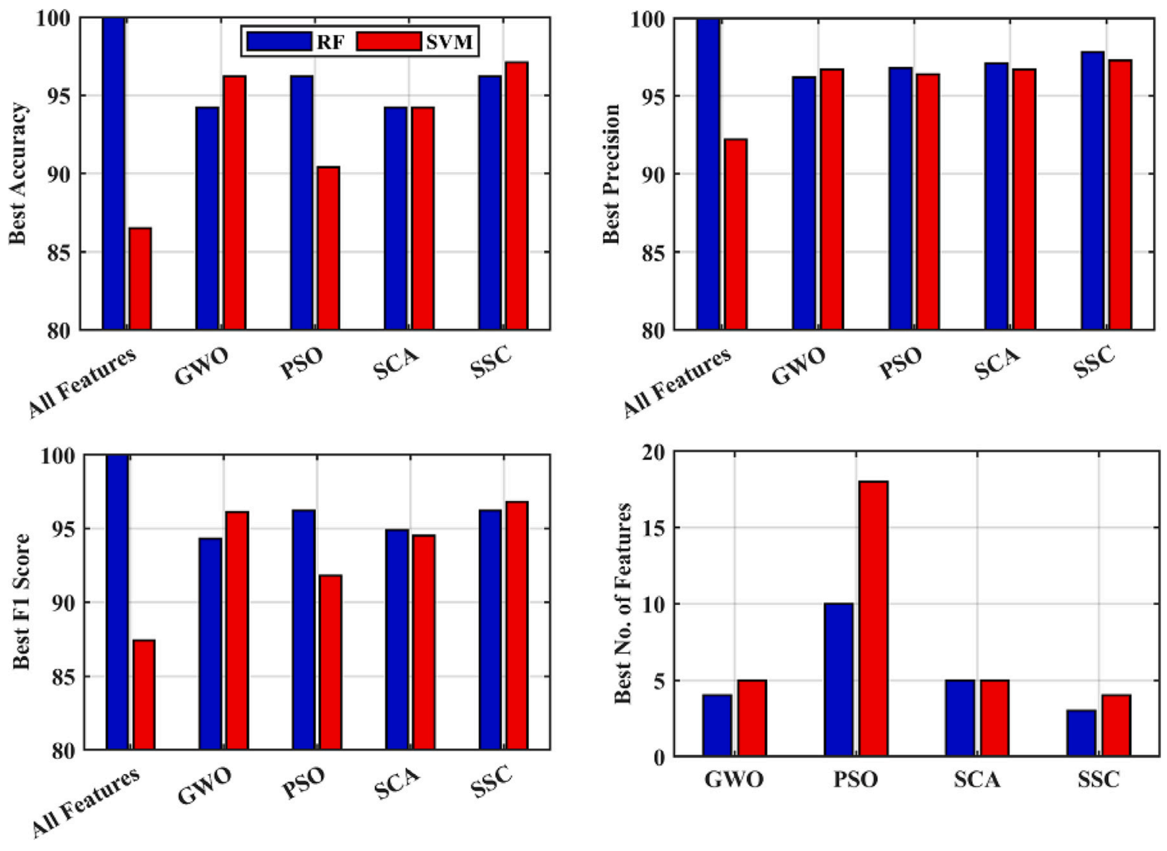
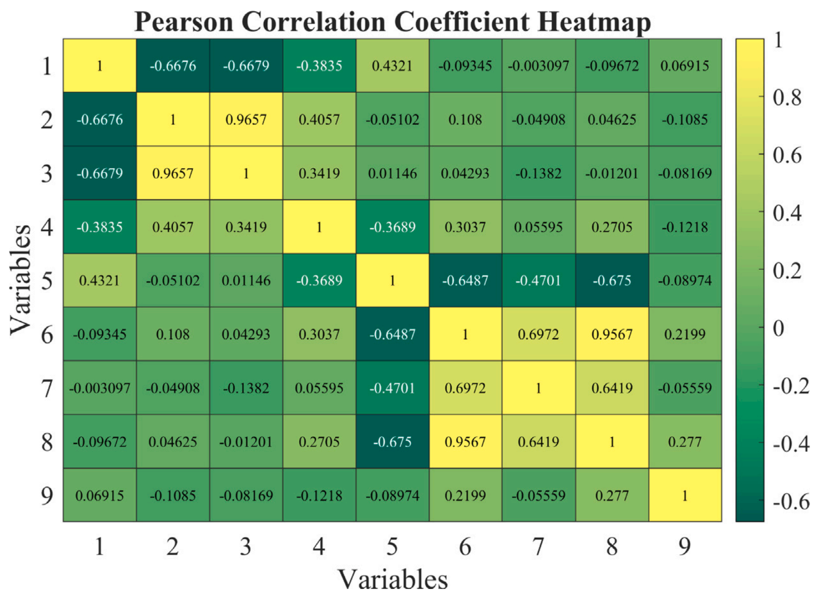


Fig. 14. Feature Selection Results: Best Accuracy, Precision, F1-Score, No. of Features.



	1	2	3	4	5	6	7	8	9
1	1	-0.6676	-0.6679	-0.3835	0.4321	-0.09345	-0.003097	-0.09672	0.06915
2	-0.6676	1	0.9657	0.4057	-0.05102	0.108	-0.04908	0.04625	-0.1085
3	-0.6679	0.9657	1	0.3419	0.01146	0.04293	-0.1382	-0.01201	-0.08169
4	-0.3835	0.4057	0.3419	1	-0.3689	0.3037	0.05595	0.2705	-0.1218
5	0.4321	-0.05102	0.01146	-0.3689	1	-0.6487	-0.4701	-0.675	-0.08974
6	-0.09345	0.108	0.04293	0.3037	-0.6487	1	0.6972	0.9567	0.2199
7	-0.003097	-0.04908	-0.1382	0.05595	-0.4701	0.6972	1	0.6419	-0.05559
8	-0.09672	0.04625	-0.01201	0.2705	-0.675	0.9567	0.6419	1	0.277
9	0.06915	-0.1085	-0.08169	-0.1218	-0.08974	0.2199	-0.05559	0.277	1

Fig. 15. Correlation of Input features and Target variable of the selected data-set.

Furthermore, we extended our research by applying the SSC algorithm to train a DNN for gesture classification. The results of our experiments with the DNN trained using SSC surpassed those obtained with other algorithms, achieving an impressive 98.7% accuracy rate. This outstanding performance further reinforces the effectiveness of the SSC algorithm for optimizing the training process of complex machine-learning models.

The findings of this research have important implications for the field of gesture recognition and related applications, such as virtual reality, augmented reality, gaming, and computer-assisted interactions. By leveraging the SSC algorithm, developers and researchers can achieve improved accuracy and efficiency in gesture recognition systems, leading to enhanced user experiences and more reliable human–computer interactions.

Exploring the adaptability of the SSC algorithm across various datasets and domains, as well as its integration with advanced machine learning architectures, could yield further insights and refinements. Scaling up the algorithm for real-time implementation and testing in

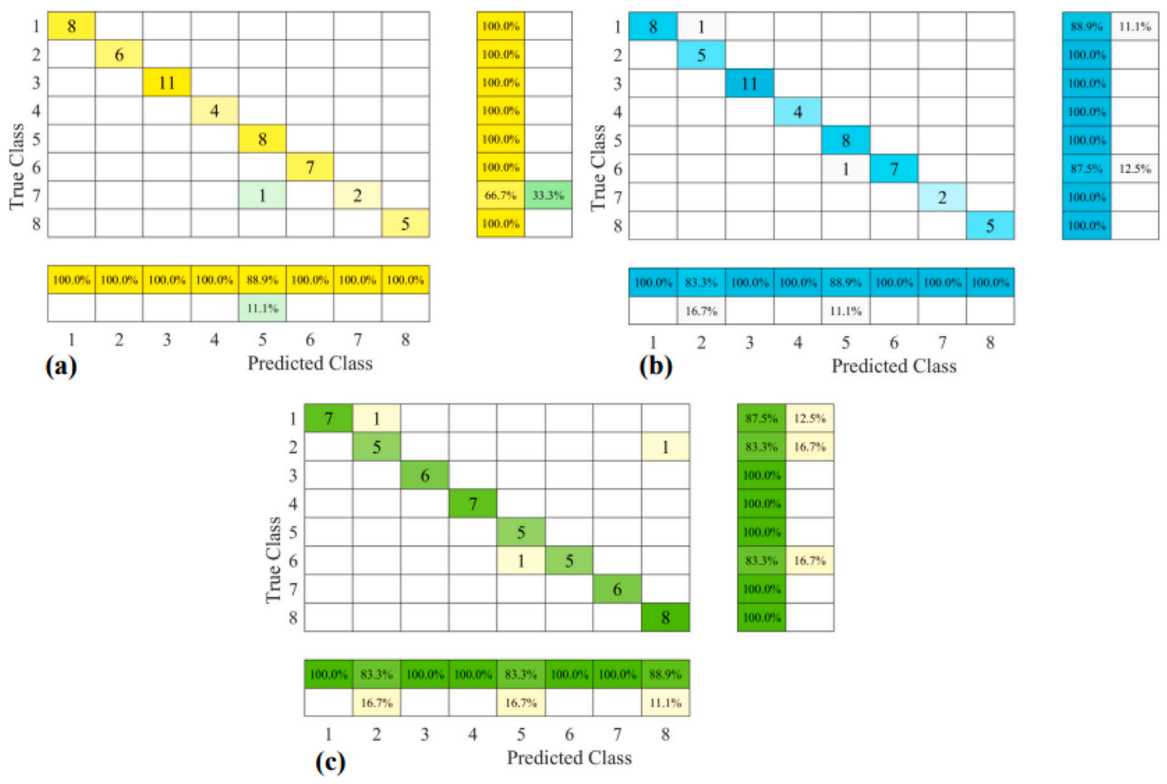


Fig. 16. Confusion Matrix of Classification Results with Selected Features.

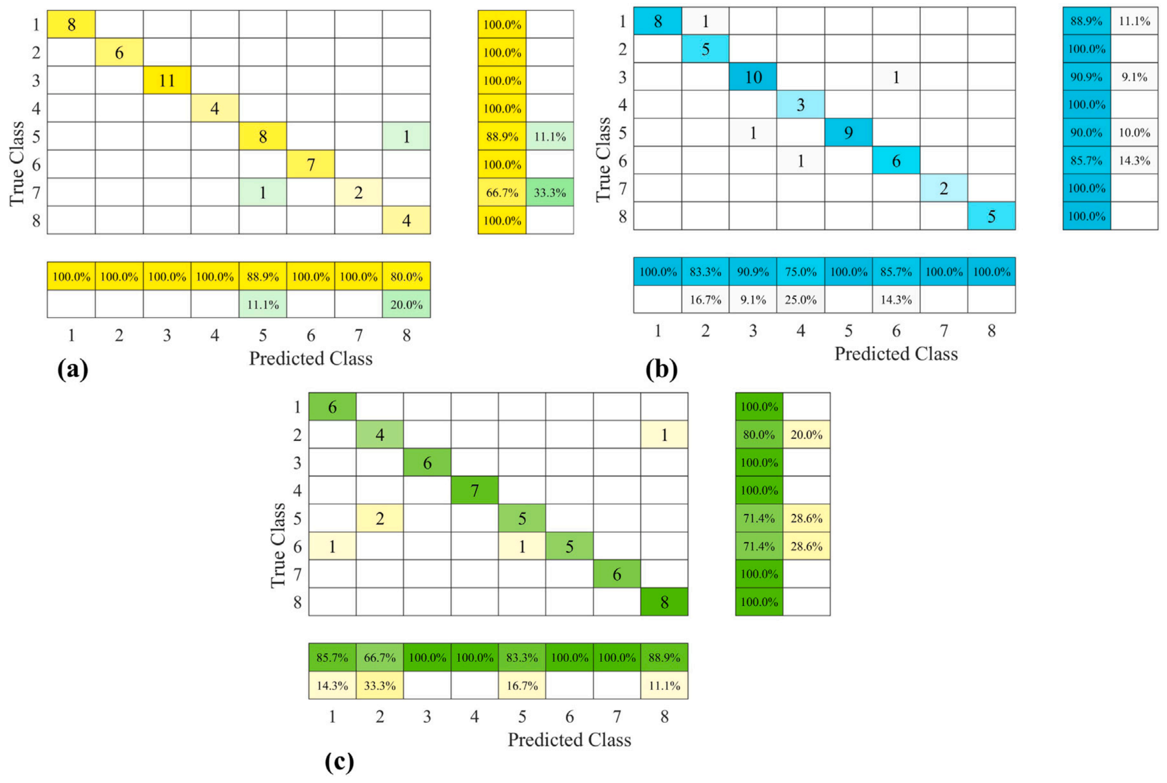


Fig. 17. Confusion Matrix of Classification Results with Full Features.

dynamic environments could bridge the gap between research and practical application. These directions hold the potential to elevate gesture recognition technology's accuracy and usability

CRediT authorship contribution statement

Hassan Mohyuddin: Conceptualization, Methodology, Resources, Project administration, Validation, Data curation, Software, Preparation and review of the original manuscript. **Syed Kumayl Raza Moosavi:** Conceptualization, Formal analysis, Investigation, Preparation and review of the original manuscript. **Muhammad Hamza Zafar:** Visualization, Data curation, Validation, Formal analysis, Preparation and review of the original manuscript. **Filippo Sanfilippo:** Supervision, Funding, acquisition, Investigation, Preparation and review of the original manuscript.

Declaration of competing interest

NONE.

All authors claim that there is not any conflict of interest regarding the above submission. The work of this submission has not been published previously. It is not under consideration for publication elsewhere. Its publication is approved by all authors and that, if accepted, it will not be published elsewhere in the same form, in English or in any other language, including electronically without the written consent of the copyright-holder.

Data availability

Data will be made available on request.

References

[1] Kambatla K, Kollias G, Kumar V, Grama A. Trends in big data analytics. J Parallel Distrib Comput 2014;74(7):2561–73. http://dx.doi.org/10.1016/j.jpdc.2014.01.003, URL https://www.sciencedirect.com/science/article/pii/S0743731514000057. Special Issue on Perspectives on Parallel and Distributed Processing.

[2] Russom P, et al. Big data analytics. TDWI Best Pract Rep 2011;19(4):1–34.

[3] Madden S. From databases to big data. IEEE Internet Comput 2012;16(3):4–6. http://dx.doi.org/10.1109/MIC.2012.50.

[4] Singh D, Reddy CK. A survey on platforms for big data analytics. J Big Data 2014;2(1):8. http://dx.doi.org/10.1186/s40537-014-0008-6.

[5] Wang RY, Strong DM. Beyond accuracy: What data quality means to data consumers. J Manage Inf Syst 1996;12(4):5–33. http://dx.doi.org/10.1080/07421222.1996.11518099.

[6] Samitsch C. Introduction. Wiesbaden: Springer Fachmedien Wiesbaden; 2015, p. 1–3.

[7] Obermeyer Z, Emanuel EJ. Predicting the future-big data, machine learning, and clinical medicine. N Engl J Med 2016;375(13):1216–9.

[8] Sun Y, Shi Y, Zhang Z. Finance big data: Management, analysis, and applications. Int J Electron Commer 2019;23(1):9–11. http://dx.doi.org/10.1080/10864415.2018.1512270.

[9] Hassanpour R, Wong S, Shahbahrami A. Visionbased hand gesture recognition for human computer interaction: A review. In: IADIS international conference interfaces and human computer interaction, Vol. 125. Citeseer; 2008.

[10] Metcalf CD, Robinson R, Malpass AJ, Bogle TP, Dell TA, Harris C, Demain SH. Markerless motion capture and measurement of hand kinematics: validation and application to home-based upper limb rehabilitation. IEEE Trans Biomed Eng 2013;60(8):2184–92.

[11] Wu C, Shahabi C. Vision-based hand gesture recognition for human-vehicle interaction: A review. IEEE Trans Intell Transp Syst 2018;20(4):1437–55.

[12] Toalumbo B, Nogales R. Hand gesture recognition using leap motion controller, infrared information, and deep learning framework. 2022, p. 412–26.

[13] Ke Q, Liu J, Bennamoun M, An S, Sohel F, Boussaid F. Chapter 5 - computer vision for human–machine interaction. In: Leo M, Farinella GM, editors. Computer vision for assistive healthcare. Computer vision and pattern recognition, Academic Press; 2018, p. 127–45. http://dx.doi.org/10.1016/B978-0-12-813445-0.00005-8, URL https://www.sciencedirect.com/science/article/pii/B9780128134450000058.

[14] Mantecón T, del-Blanco CR, Jaureguizar F, García N. Hand gesture recognition using infrared imagery provided by leap motion controller. In: Blanc-Talon J, Distante C, Philips W, Popescu D, Scheunders P, editors. Advanced concepts for intelligent vision systems. Cham: Springer International Publishing; 2016, p. 47–57.

[15] Yang Q, Ding W, Zhou X, Zhao D, Yan S. Leap motion hand gesture recognition based on deep neural network. In: 2020 chinese control and decision conference (CCDC). 2020, p. 2089–93. http://dx.doi.org/10.1109/CCDC49329.2020.9164723.

[16] Nogales R, Benalcázar M. Real-time hand gesture recognition using the leap motion controller and machine learning. In: 2019 IEEE latin american conference on computational intelligence (la-CCI). 2019, p. 1–7. http://dx.doi.org/10.1109/LA-CCI47412.2019.9037037.

[17] Chandrashekar G, Sahin F. A survey on feature selection methods. Comput Electr Eng 2014;40(1):16–28.

[18] Dash M, Liu H. Feature selection for classification. Intell Data Anal 1997;1(3):131–56.

[19] Brownlee J. Feature selection with the caret r package. 2016.

[20] Jolliffe IT. Principal component analysis. John Wiley & Sons; 2002.

[21] Belhumeur PN, Hespanha JP, Kriegman DJ. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. IEEE Trans Pattern Anal Mach Intell 1997;19(7):711–20.

[22] Guzsvinecz T, Szucs V, Sik-Lanyi C. Suitability of the kinect sensor and leap motion controller—A literature review. Sensors 2019;19(5):1072.

[23] Sanfilippo F, Pacchierotti C. A low-cost multi-modal auditory-visual-tactile framework for remote touch. In: Proc. of the IEEE 3rd international conference on information and computer technologies (ICICT). 2020, p. 213–8.

[24] Sanfilippo F, Hatledal LI, Pettersen K. A fully–immersive hapto–audio–visual framework for remote touch. In: Proc. of the 11th IEEE international conference on innovations in information technology (IIT'15), Dubai, United Arab Emirates. 2015.

[25] Sanfilippo F, Blazauskas T, Salvietti G, Ramos I, Vert S, Radianti J, Majchrzak TA, Oliveira D. A perspective review on integrating VR/AR with haptics into stem education for multi-sensory learning. Robotics 2022;11(2):41.

[26] Sanfilippo F, Blažauskas T, Girdžiūna M, Janonis A, Kiudys E, Salvietti G. A multi-modal auditory-visual-tactile e-learning framework. In: Proc. of the 4th international conference on intelligent technologies and applications, INTAP 2021, Grimstad, Norway, October 11–13, 2021, Revised Selected Papers. Springer; 2022, p. 119–31.

[27] Weichert F, Bachmann D, Rudak B, Fisseler D. Analysis of the accuracy and robustness of the leap motion controller. Sensors 2013;13(5):6380–93. http://dx.doi.org/10.3390/s130506380, URL https://www.mdpi.com/1424-8220/13/5/6380.

[28] Rangarajan L, et al. Bi-level dimensionality reduction methods using feature selection and feature extraction. Int J Comput Appl 2010;4(2):33–8.

[29] Motoda H, Liu H. Feature selection, extraction and construction. Commun IICM (Inst Inf Comput Mach Taiwan) 2002;5(67–72):2.

[30] Agrawal P, Abutarboush HF, Ganesh T, Mohamed AW. Metaheuristic algorithms on feature selection: A survey of one decade of research (2009–2019). IEEE Access 2021;9:26766–91. http://dx.doi.org/10.1109/ACCESS.2021.3056407.

[31] Gendreau M, Potvin J-Y. Metaheuristics in combinatorial optimization. Ann Oper Res 2005;140(1):189–213. http://dx.doi.org/10.1007/s10479-005-3971-7.

[32] Dokeroglu T, Sevinc E, Kucukyilmaz T, Cosar A. A survey on new generation metaheuristic algorithms. Comput Ind Eng 2019;137:106040.

[33] Alba E, Dorronsoro B. The exploration/exploitation tradeoff in dynamic cellular genetic algorithms. IEEE Trans Evol Comput 2005;9(2):126–42.

[34] Olorunda O, Engelbrecht AP. Measuring exploration/exploitation in particle swarms using swarm diversity. In: 2008 IEEE congress on evolutionary computation (IEEE world congress on computational intelligence). IEEE; 2008, p. 1128–34.

[35] Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of ICNN'95-international conference on neural networks, Vol. 4. IEEE; 1995, p. 1942–8.

[36] Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. Adv Eng Softw 2014;69:46–61.

[37] Karaboga D. Artificial bee colony algorithm. Scholarpedia 2010;5(3):6915.

[38] Mirjalili S, Lewis A. The whale optimization algorithm. Adv Eng Softw 2016;95:51–67.

[39] Blum C, Puchinger J, Raidl GR, Roli A. Hybrid metaheuristics in combinatorial optimization: A survey. Appl Soft Comput 2011;11(6):4135–51. http://dx.doi.org/10.1016/j.asoc.2011.02.032.

[40] Azzougui Y, Recioui A, Mansouri A. PMU optimal placement in wide area monitoring systems using grey wolf optimization technique. 2019;4:1–7 http://dx.doi.org/10.51485/ajss.v4i1.76 URL https://ajss.dz/index.php/ajss/article/view/76.

[41] Zhang X, Chen X, Li Y, Lantz V, Wang K, Yang J. A framework for hand gesture recognition based on accelerometer and EMG sensors. IEEE Trans Syst Man Cybern A 2011;41(6):1064–76.

[42] Hoggan E, Williamson J, Oulasvirta A, Nacenta M, Kristensson PO, Lehtiö A. Multi-touch rotation gestures: Performance and ergonomics. In: Proceedings of the sigchi conference on human factors in computing systems. 2013, p. 3047–50.

[43] Kammer D, Wojdziak J, Keck M, Groh R, Taranko S. Towards a formalization of multi-touch gestures. In: ACM international conference on interactive tabletops and surfaces. 2010, p. 49–58.

[44] Sharma JK, Gupta R, Pathak VK. Numeral gesture recognition using leap motion sensor. In: 2015 international conference on computational intelligence and communication networks (CICN). 2015, p. 411–4. http://dx.doi.org/10.1109/CICN.2015.86.

[45] Vaitkevičius A, Taroza M, Blažauskas T, Damaševičius R, Maskeliūnas R, Woźniak M. Recognition of American sign language gestures in a virtual reality using leap motion. Appl Sci 2019;9(3). http://dx.doi.org/10.3390/app9030445, URL https://www.mdpi.com/2076-3417/9/3/445.

[46] Yao Y, Fu Y. Contour model-based hand-gesture recognition using the kinect sensor. IEEE Trans Circuits Syst Video Technol 2014;24(11):1935–44. http://dx.doi.org/10.1109/TCSVT.2014.2302538.

[47] Zeng W, Wang C, Wang Q. Hand gesture recognition using leap motion via deterministic learning. Multimed Tools Appl 2018;77:28185–206.

[48] Lu W, Tong Z, Chu J. Dynamic hand gesture recognition with leap motion controller. IEEE Signal Process Lett 2016;23(9):1188–92. http://dx.doi.org/10.1109/LSP.2016.2590470.

[49] Xu Y, Wang Q, Bai X, Chen Y-L, Wu X. A novel feature extracting method for dynamic gesture recognition based on support vector machine. In: 2014 IEEE international conference on information and automation (ICIA). 2014, p. 437–41. http://dx.doi.org/10.1109/ICInfA.2014.6932695.

[50] Xue B, Zhang M, Browne WN, Yao X. A survey on evolutionary computation approaches to feature selection. IEEE Trans Evol Comput 2015;20(4):606–26.

[51] Akinola OO, Ezugwu AE, Agushaka JO, Zitar RA, Abualigah L. Multiclass feature selection with metaheuristic optimization algorithms: a review. Neural Comput Appl 2022;34(22):19751–90.

[52] Chen L, Liu Y, Liu X. Gesture recognition based on improved particle swarm optimization feature selection and improved support vector machine. In: Proceedings of the 2013 IEEE international conference on robotics and automation (ICRA). 2013, p. 2201–6.

[53] Wang J, Khishe M, Kaveh M, Mohammadi H. Binary chimp optimization algorithm (BChOA): a new binary meta-heuristic for solving optimization problems. Cogn Comput 2021;13:1297–316.

[54] Sayed GI, Tharwat A, Hassanien AE. Chaotic dragonfly algorithm: an improved metaheuristic algorithm for feature selection. Appl Intell 2019;49:188–205.

[55] SSC: A hybrid nature-inspired meta-heuristic optimization algorithm for engineering applications. Knowl Based Syst 2021;222:106926. http://dx.doi.org/10.1016/J.KNOSYS.2021.106926.

[56] Mirjalili S. SCA: A Sine cosine algorithm for solving optimization problems. Knowl-Based Syst 2016;96:120–33. http://dx.doi.org/10.1016/j.knosys.2015.12.022, URL https://www.sciencedirect.com/science/article/pii/S0950705115005043.

[57] Dhiman G, Kumar V. Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications. Adv Eng Softw 2017;114:48–70. http://dx.doi.org/10.1016/j.advengsoft.2017.05.014, URL https://www.sciencedirect.com/science/article/pii/S0965997816305567.

[58] Khishe M, Mosavi M. Chimp optimization algorithm. Expert Syst Appl 2020;149:113338. http://dx.doi.org/10.1016/j.eswa.2020.113338, URL https://www.sciencedirect.com/science/article/pii/S0957417420301639.

[59] Mirjalili S. SCA: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 2016;96:120–33.

[60] Kowdiki M, Khaparde A. Automatic hand gesture recognition using hybrid meta-heuristic-based feature selection and classification with dynamic time warping. Comp Sci Rev 2021;39:100320. http://dx.doi.org/10.1016/j.cosrev.2020.100320, URL https://www.sciencedirect.com/science/article/pii/S1574013720304202.

[61] Dubey AK. Enhanced hand-gesture recognition by improved beetle swarm optimized probabilistic neural network for human–computer interaction. J Ambient Intell Humaniz Comput 2022.

[62] Zeng W, Wang C, Wang Q. Hand gesture recognition using leap motion via deterministic learning. Multimed Tools Appl 2018;77:28185–206.

[63] Hisham B, Hamouda A. Arabic sign language recognition using ada-boosting based on a leap motion controller. Int J Inf Technol 2021;13:1221–34.

[64] Rakesh S, Kovács G, Mokayed H, Saini R, Pal U. Static palm sign gesture recognition with leap motion and genetic algorithm. In: 2021 swedish artificial intelligence society workshop (SAIS). 2021, p. 1–5. http://dx.doi.org/10.1109/SAIS53221.2021.9508468.

[65] Chong T-W, Lee B-G. American sign language recognition using leap motion controller with machine learning approach. Sensors 2018;18(10):3554.

[66] Feng K-p, Yuan F. Static hand gesture recognition based on HOG characters and support vector machines. In: 2013 2nd international symposium on instrumentation and measurement, sensor network and automation (IMSNA). 2013, p. 936–8. http://dx.doi.org/10.1109/IMSNA.2013.6743432.