Konya Mühendislik Bilimleri Dergisi., c. 13, s. 4, 1137-1157, 2025
Konya Journal of Engineering Sciences v. 13, n. 4, 1137-1157, 2025
ISSN: 2667-8055 (Electronic)
DOI: 10.36306/konjes.1679144

THE STUDY OF THE HYPERPARAMETER OPTIMIZATION ANALYSIS IN THE CONVOLUTIONAL NEURAL NETWORKS MODEL WITH STARFISH OPTIMIZATION ALGORITHM

[1,*] Gülnur YILDIZDAN, [2] Emine BAS

[1] Selcuk University, Kulu Vocational School, Konya, TÜRKİYE
[2] Konya Technical University, Computer and Information Sciences Faculty, Software Engineering Department, Konya, TÜRKİYE
[1] gavsar@selcuk.edu.tr, [2] ebas@ktun.edu.tr

Highlights

- CNN training parameters and network layers were optimized for the highest accuracy.
- Hyperparameter optimization was performed on the CNN model with SFOA.
- Multiple hyperparameters were optimized simultaneously.
- The proposed model was tested on MNIST, Kuzushiji-MNIST and EMNIST datasets.
- The proposed model achieved 99.52%, 97.91% and ~88% accuracy on these datasets.

***Corresponding Author:** Gülnur YILDIZDAN, gavsar@selcuk.edu.tr

Konya Mühendislik Bilimleri Dergisi., c. 13, s. 4, 1137-1157, 2025
Konya Journal of Engineering Sciences v. 13, n. 4, 1137-1157, 2025
ISSN: 2667-8055 (Electronic)
DOI: 10.36306/konjes.1679144

THE STUDY OF THE HYPERPARAMETER OPTIMIZATION ANALYSIS IN THE CONVOLUTIONAL NEURAL NETWORKS MODEL WITH STARFISH OPTIMIZATION ALGORITHM

[1,*] Gülnur YILDIZDAN, [2] Emine BAS

[1] *Selcuk University, Kulu Vocational School, Konya, TÜRKİYE*
[2] *Konya Technical University, Computer and Information Sciences Faculty, Software Engineering Department, Konya, TÜRKİYE*
[1] gavsar@selcuk.edu.tr, [2] ebas@ktun.edu.tr

(Received: 18.04.2025; Accepted in Revised Form: 25.07.2025)

ABSTRACT: In this study, the hyperparameters of Convolutional Neural Networks (CNNs) have been optimized with the newly proposed Starfish Optimization Algorithm (SFOA) in recent years. CNN has complex hyperparameters due to its structure. In the literature, the values of hyperparameters are mostly tried to be determined with combinatorial methods. The success of metaheuristic algorithms in optimizing the variables of different problems has inspired this study. Thus, four different numbers of channel values (8, 16, 32, and 64), five different kernel size values (1×1, 3×3, 5×5, 7×7, and 9×9), four different batch size values (32, 64, 128, and 256), twenty different values randomly generated between 0 and 0.05 for the learning rate, three different optimizer types (sgdm, adam, and rmsprop), and four different epoch values (5, 10, 15, and 20), which are the most critical hyperparameters in CNN, have been determined. A 6-dimensional solution space was determined with SFOA, and these hyperparameter values were placed in discretely defined dimensions. SFOA tried to determine the most appropriate hyperparameter values for the CNN model in each iteration. In this study, two different image datasets (MNIST and Kuzushiji-MNIST) were selected for CNN classification. Due to the hyperparameter optimization carried out with the SFOA algorithm, an accuracy of 99.52% for the MNIST dataset and 97.91% for the Kuzushiji-MNIST dataset was achieved. Comparisons with existing literature demonstrate that the proposed model showcases successful and competitive performance. Finally, the proposed CNN models are evaluated on a different image dataset, EMNIST (Extended MNIST). EMNIST is a more comprehensive version of MNIST developed for classifying handwritten letters and numbers. The accuracy results on the EMNIST dataset were 88.65% (the proposed CNN model with similar hyperparameter settings as MNIST) and 88.73% (the proposed CNN model with similar hyperparameter settings as Kuzushiji-MNIST), respectively. Additionally, hyperparameters for the EMNIST dataset were determined using SFOA, achieving an accuracy of 88.71%. Analyzing the hyperparameters of three different CNN models, it was observed that similar optimizer types, epoch numbers, kernel sizes, and channel numbers were preferred. This demonstrates that SFOA can produce reliable and effective settings across different datasets.

Keywords: *Convolutional Neural Networks, Hyperparameter Analysis, Kuzushiji-MNIST, MNIST, EMNIST, SFOA*

1. INTRODUCTION

One kind of multilayer artificial neural network that uses convolution instead of matrix multiplication in one of its layers is called a Convolutional Neural Network (CNN). CNN and classical Artificial Neural Networks (ANNs) have different structures. CNNs, unlike ANNs, perform data processing and feature extraction by navigating over inputs with a matrix (filter/kernel) of predetermined square size. In ANNs, the system is first trained with examples, then the error value of the system is determined by testing. CNN achieves very successful results especially in image classification, so it is used by many researchers in image classification. Despite this, there are still many problems to be overcome in CNN. In studies to increase the success of CNN results, CNN architectures are developed more deeply and more

***Corresponding Author:** Gülnur YILDIZDAN, gavsar@selcuk.edu.tr

computational costs arise. Reducing these computational costs is directly dependent on the hyperparameter values selected in the CNN architecture. Choosing the right hyperparameters in the CNN model is of great importance [1], [2].

Researchers generally try to determine the most successful values of a few selected hyperparameters in CNN with probabilistic methods. Optimization of all hyperparameters requires high processing power. In such an experiment, evaluating all possible possibilities specified for hyperparameter values (grid search) is very difficult in practical terms. In recent years, researchers have been using metaheuristic algorithms in hyper-parameter optimization in CNN architecture due to their success in detecting problem variables. Bergstra et al. examined different hyperparameter optimization algorithms [3]. These methods are grid search, random search, Bayesian optimization and sequential model-based optimization (SMBO). Information about the strengths, weaknesses and usage scenarios of the methods is presented [3]. Gülcü and Kuş examined the studies that performed hyperparameter optimization of CNN with methods such as Genetic Algorithm, Particle Swarm Optimization, Differential Evolution and Bayesian Optimization. In their studies, the optimized hyperparameters, defined value ranges and obtained results were analyzed. According to their studies, it was determined that the most effective hyperparameters in the performance of CNN are the number of filters, filter size, number of layers, sparsification rate, learning rate and batch size [1]. Lindauer et al. have developed SMAC3, a powerful and adaptive framework for Bayesian Optimization. This model allows users to quickly improve performance by evaluating a minimum number of configurations, allowing them to efficiently determine the hyperparameter settings that yield the best results for their algorithm, dataset, and application [4]. Li et al., in their study, have conducted a study on Hyperband, a bandit-based hyperparameter optimization method. Hyperband aims to find the best performing hyperparameters by quickly eliminating a set of random configurations. The study discusses the effectiveness, efficiency and performance of the Hyperband approach in different application scenarios [5]. Khanday et al. tried to determine the hyperparameter of filter size in image classification using CIFAR10 and FashionMNIST datasets. They examined the effects of various filter sizes on the accuracy of the models. The results of the experiment show that the accuracy decreases as the filter size increases [6]. Shao et al. developed a CNN model to improve the classification rate of MNIST handwritten digit dataset. The developed model includes a multilayer deep learning structure consisting of three convolution and activation layers to facilitate feature extraction. In addition, two fully connected layers are included to enable classification. The hyperparameters of the model are optimized to improve the classification performance. The optimized hyperparameters include batch sizes, kernel sizes, batch normalization, activation function, and learning rate [7]. Bischl et al. provided recommendations on important choices to make when conducting hyperparameter optimization, including hyperparameter performance evaluation, how to combine hyperparameter optimization with machine learning pipelines, runtime optimizations, and parallelization [8]. Yiğit et al. used convolutional neural network for recognition of handwritten digits using MNIST dataset. Hyper parameters in CNN model were analyzed using Optuna, HyperOpt and Scikit-optimize libraries and the results were evaluated. Optimization times for hyper parameter libraries and changes in success rate in recognition of handwritten digits were analyzed [9]. Orhan et al. optimized the parameters used in CNN training and the layers used in creating the network structure in order to select the hyperparameters that give the highest accuracy value. The determined hyperparameters are the number of channels, convolutional layers, minimum batch size, activation function, and learning rate. In their work, they used Distributed Search (DA) and Genetic Algorithms (GA) methods to perform hyperparameter optimization. It was observed that the DA method determined more appropriate hyperparameter values than GA [10]. Xiao et al. performed hyperparameter optimization using variable length GA on the CIFAR-10 dataset [11]. Tanyıldızı and Demirtaş examined four methods (grid search, random search, Bayesian and evolutionary algorithms) that are frequently used in the literature in hyperparameter analysis. In their study, these methods were discussed and their advantages and disadvantages were determined [12]. Özbay et al. performed skin cancer classification with the ISIC-2019 dataset. They optimized the hyper-parameters in the proposed CNN model with the PSO algorithm in their study [13]. İnik is defined a CNN architecture using Particle Swarm Optimization

(PSO) algorithm. Thus, hyper parameters in CNN are fully represented without any transformation during optimization. CNN architecture studies were carried out on ESC-10, ESC-50 and Urbansound8k audio datasets [14]. In Kıymaç's study, a CNN model was developed for the classification of ECG signals. In the CNN model, Relu was used in the hidden layers and softmax in the output layer as the activation function. The Adam algorithm was preferred as the optimizer. The created artificial neural network was trained with 10 epochs. Different discrete values were determined for hyper-parameters such as the batch size to be optimized, learning rate, convolutional layer and sparsity rate, and PSO was determined [15]. Yurdakul were determined the number and size of filters, activation function, pooling algorithm, fully connected layer number, number of neurons and learning rate hyper parameters by meta-heuristic algorithms such as artificial bee colony algorithm. CIFAR-10 dataset was used in the study. In study, Yurdakul determined the optimum value ranges for hyper parameters as; between 5 and 9 for filter size, between 64 and 256 for filter number, between 32 and 256 for neuron number and between 0.5 and 0.7 for sparsification ratio [16]. Atteia et al. optimized the hyper parameters in the CNN model used in the diagnosis of diabetic maculopathy. In the study, the Bayesian optimization method was used to optimize the CNN hyper parameters. According to the results, it was shown that the CNN model obtained with Bayesian optimization performed better than a non-optimized network. In the study, hyper parameters such as learning rate, number of neurons in layers, activation function and mini-batch size were optimized [17]. Amou et al. performed the classification of brain tumor with CNN and optimized CNN parameters with Bayesian optimization. The optimized hyperparameters are activation function, minimum batch size, sparsity ratio, dense layer node count, gradient descent optimization algorithm [18]. ZainEldin et al. performed a classification using CNN in their study. In the study, they proposed a hybrid heuristic algorithm and optimized many parameters such as learning rate, momentum rate, minimum batch size, L2 regularization rate, and number of cycles, which are the hyper parameters of CNN, with this hybrid heuristic algorithm [19]. Sethi et al. conducted a study using CNN and LSTN models for early diagnosis of Alzheimer's disease. In their study, they tried to detect hyperparameter values using Bayesian optimization method. The hyperparameters they tried to detect are learning rate, optimization algorithms, batch size, activation functions and regularization parameters (L1, L2) [20]. Zhang et al. performed lung module classification using CNN. In their study, a gaussian surrogate model was used for hyperparameter optimization. Parameters such as learning rate, batch size, and epoch number were optimized in the study [21]. Shankar et al. conducted a diabetic retinopathy detection study. In their study, they used Bayesian optimization algorithm for hyperparameter optimization. In this method, they used a "surrogate" model to facilitate the optimization of the target function [22]. Seng et al. analyzed the MNIST dataset to train the CNN ResNet-18 model to recognize handwritten digits accurately. Five different models were used in the study: GoogLeNet, ResNet-50, ResNeXt-50, MobileNet v2 and Wide ResNet-50 [23].

When the literature is examined, it is seen that Bayesian algorithms are mostly preferred in hyperparameter optimization for deep network models. It is seen that metaheuristic algorithms are also used in some studies. It is observed that PSO and GA algorithms are especially preferred. Today, there are metaheuristic algorithms that are much more effective and have the ability to systematically exploration and exploitation the search space compared to PSO and GA. This study was carried out to fill a gap in the literature due to the lack of use of such metaheuristic algorithms. In this study, the Starfish Optimization Algorithm (SFOA), which has been newly proposed in recent years, was used to determine the hyperparameters of the CNNs model. The SFOA algorithm was proposed by Zhong et al. in 2025 [24]. Since this algorithm is a newly proposed algorithm, there is a published study on SFOA in the literature at the time this study was prepared. Izci et al. developed a two-degree-of-freedom PID acceleration (2DOF-PIDA) controller with SFOA [25]. Apart from this, no other research in the literature uses SFOA. In this study, six different hyperparameters (channel number, kernel size, batch size, learning rate, optimizer type, and epoch size) are determined for hyperparameter optimization. These hyperparameters are determined in a CNN model on MNIST and Kuzushiji-MNIST datasets. The MNIST dataset consists of 70,000 black-and-white images of handwritten digits, 28×28 pixels in size. These images represent the digits 0 to 9. The MNIST dataset contains different handwriting styles within each digit class. That is, there

are examples of the same digit written in different ways by different people [10]. MNIST is one of the widely used datasets in the literature [6], [7], [9], [10], [23]. There are many studies in the literature that use CNN in the classification problem of the MNIST dataset and optimize these CNN hyper-parameters. Kadam et al. (2020) performed a classification on MNIST and FashionMNIST image data using five different CNN architectures with varying convolutional layers, filter size, and fully connected layers. In these architectures, they varied the hyperparameters of the CNN structure, namely activation function, optimizer, learning rate, dropout rate, and batch size. The results confirmed that the choices of activation function, optimizer and dropout rate have the most impact on the accuracy [26]. Shao et al. (2023) proposed a deep CNN model to classify MNIST image data. The proposed model includes 3 convolution and activation layers for feature extraction and 2 fully connected layers for classification. The hyperparameters such as batch size, kernel size, batch normalization, activation function, and learning rate in the CNN model are optimized by the authors to improve the recognition performance [7]. Ahlawat et al. (2020) proposed a CNN based handwritten digit recognition system. MNIST handwritten digits were used for testing purposes. Various design options such as number of layers, stride size, receptive field, kernel size, padding, and sparsification were optimized in their proposed CNN model. The effects of different SGD optimization algorithms on performance were also investigated [27]. They analyzed the performance of the deep neural network based Convolutional Neural Network model on the MNIST dataset. Jain et al. (2018) analyzed the values of the parameters of CNN, such as error function, activation function, number of hidden layers, number of epochs, convex and non-convex optimized objective function, through various tests [28]. Yahya et al. (2021) determined the most suitable CNN model by considering the appropriate filter size selection, data preparation, limitations in datasets and noise. The performance of this model was tested on the MNIST dataset [29]. Gilanie et al. (2025) proposed parameter optimization of an autocoder for image classification using genetic algorithm (GA). Autocoder is a neural network architecture commonly used for unsupervised learning, dimensionality reduction, and feature extraction. Its performance largely depends on hyperparameters that need to be carefully tuned to obtain optimal results. In their work, they proposed a GA-based optimization approach to fine-tune the hyperparameters of an autocoder, including the number of hidden layers, number of neurons per layer, activation function, learning rate, and batch size. The proposed approach is tested on MNIST, EMNIST, and Fashion-MNIST [30]. Kılıçarslan (2023) proposed particle swarm optimization (PSO), cat swarm optimization (CSO) and a hybrid algorithm of PSO and gray wolf optimization (GWO) for the optimization of hyperparameters of 1D-VGG-16 model. The performance of 1D-VGG-16 model with the most suitable hyperparameters was tested on cardiovascular disease data. Hyper-parameter optimization of 2D-VGG-16 architecture was performed with PSO, CSO and PSO + GWO hybrid optimization algorithms using MNIST dataset [31]. Kuzushiji-MNIST is a dataset featuring traditional Kuzushiji handwritten characters from Japan. This dataset resembles the MNIST dataset of written figures but consists of Japanese handwritten characters. Kuzushiji-MNIST includes a test set of 70000 samples from the traditional handwritten characters database. Kuzushiji-MNIST is often used in the literature, though not as frequently as MNIST [32], [33].

The primary contributions of the study are as follows:

- In this study, the Starfish Optimization Algorithm (SFOA) has been comprehensively evaluated for the first time in the optimization of deep learning hyperparameters, despite its limited use in the literature.
- It was tested on both Latin and Japanese character-based data sets (MNIST and Kuzushiji-MNIST), and the general validity of the method in different data types was demonstrated.
- In addition to the accuracy rate, model performance was measured with metrics such as precision, recall, and F1 score, and it was revealed that the model provided not only high accuracy but also balanced and reliable performance.

- Multiple hyperparameters, such as the number of channels, kernel size, batch size, learning rate, optimizer type, and epoch number, were optimized simultaneously. A model that can perform effective searches in this wide hyperparameter space was developed and contributed to the literature.
- Many studies have been presented in the literature on MNIST and Kuzushiji-MNIST datasets using the CNN deep network classifier. However, these studies used different CNN architectures with different hyperparameter values. This study shows originality in terms of CNN architecture and the hyperparameter values detected in this architecture.
- The CNN model optimized with SFOA is among the best models in the literature, with 99.52% accuracy on the MNIST data set and 97.91% accuracy on Kuzushiji-MNIST. Thus, not only in theory but also in practice, the proposed model has proven to be a strong alternative.
- To assess the generalizability of the proposed CNN model, it was also evaluated on the EMNIST dataset. Using similar hyperparameters, MNIST and Kuzushiji-MNIST achieved accuracies of 88.65% and 88.73%, respectively, while the SFOA-optimized settings resulted in 88.71% accuracy. The consistency of the hyperparameters indicates that SFOA can produce reliable and effective settings across different datasets. These findings support the generalizability and robustness of the proposed medel.

In the Material and Method Section of the study, the details of the SFOA algorithm, the created CNN model, the dataset definitions used in the study, and the measurement metrics are explained in detail. In the Results and Discussion Section, the experimental results obtained are presented and discussed with graphics and visuals. In the Conclusion Section, the results of the study are interpreted, and future planned studies are mentioned.

2. MATERIAL AND METHODS

2.1. Starfish Optimization Algorithm (SFOA)

Starfish are marine invertebrates belonging to the echinoderms of the class Asteroidea. There are more than 2000 species in the world. Starfish usually have five arms. They live in the deep oceans. They can live for about 10 years. The starfish's eyes are located at the ends of its arms and sense environmental conditions. The starfish's mouth is located in the center of its body. Although sea stars are marine creatures, they do not have fins, gills or scales and cannot swim like fish. They do not have a brain or heart. They have tough skin and small protrusions or spines. They can be a variety of colors, such as red or purple. Sea stars are capable of both sexual and asexual reproduction. Starfish are carnivorous and can consume a variety of marine life. They sometimes prey on creatures larger than themselves. If a starfish loses any of its arms, it can grow a new one. A new starfish can also grow from a lost arm. Starfish have a remarkable ability to regenerate [34], [35], [36], [37].

2.1.1. Mathematical model of SFOA

Inspired by the exploration, hunting, and regeneration behaviors of starfish, an optimization algorithm called starfish optimization algorithm (SFOA) was proposed by Zhong et al. in 2025.

Initialization:

The positions of the starfish are randomly generated by Equations 1–2 between the variable boundaries given as a matrix [24] during the initialization phase of SFOA.

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N\times D} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,D} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ x_{1,i} & \cdots & x_{i,j} & \cdots & x_{i,D} \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ x_{N,i} & \cdots & x_{N,j} & \cdots & x_{N,D} \end{bmatrix}_{N\times D} \quad (1)$$

$$x_{i,j} = l_j + r_{i,j} \times (u_j - l_j), \quad i = 1, 2, 3, \dots, N; j = 1, 2, 3, \dots, D \tag{2}$$

where X is a matrix representing the positions of the starfish, with a size of $N \times D$, where N is the population size and D is the dimension of the problem. The term $x_{i,j}$ denotes the *jth* dimension position of the *ith* starfish, r represents a random number between (0, 1), and u_j and l_j refer to the upper and lower bounds of the problem in the *jth* dimension, respectively. Following the random generation of all starfish positions as matrix X, each starfish position is evaluated in the objective function (F). The evaluation results are shown in Equation 3 [24].

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N\times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N\times 1} \tag{3}$$

where F stores the evaluation results in a matrix of size $N \times 1$.

Exploration phase:

In this stage, SFOA modeled the exploration behavior of starfish according to their arm counts. If $D \leq 5$, a single arm is used to search in a single direction, while if $D > 5$, all five arms are used to explore the search space in all directions [24].

- If the optimization problem has a dimension larger than 5 ($D > 5$), this situation is modeled in Equations 4-7.

$$\begin{cases} Y_{i,p}^T = X_{i,p}^T + a_1\left(X_{best,p}^T - X_{i,p}^T\right) cos\ \theta, & r \leq 0.5 \\ Y_{i,p}^T = X_{i,p}^T + a_1\left(X_{best,p}^T - X_{i,p}^T\right) sin\ \theta, & r > 0.5 \end{cases} \tag{4}$$

$$a_1 = (2r - 1)\pi \tag{5}$$

$$\theta = \frac{\pi}{2} . \frac{T}{T_{max}} \tag{6}$$

$$X_{i,p}^{T+1} = \begin{cases} Y_{i,p}^T & l_{b,p} \leq Y_{i,p}^T \leq u_{b,p} \\ X_{i,p}^T & else \end{cases} \tag{7}$$

$Y_{i,p}^T$ and $X_{i,p}^T$ represent the obtained and current positions of a starfish, respectively. $X_{best,p}^T$ represents the p-dimensional vector of the current optimal position, where p consists of five randomly selected dimensions from a total of D dimensions, and r is a value within the interval (0,1). T represents the current iteration, while T_{max} indicates the maximum iteration number. The sine and cosine functions indicate that the arms of starfish can twist either left or right to access food with equal likelihood. p indicates the updated dimension, while $l_{b,p}$ and $u_{b,p}$ represent the problem's bounds, respectively [24].

- If the optimization problem does not have a dimension larger than 5 ($D > 5$), this situation is modeled in Equations 8-9.

$$Y_{i,q}^T = E_t X_{i,p}^T + A_1\left(X_{k1,p}^T - X_{i,p}^T\right) + A_2\left(X_{k2,p}^T - X_{i,p}^T\right) \tag{8}$$

$$E_t = \frac{T_{max} - T}{T_{max}} cos\ \theta \tag{9}$$

$X_{k1,p}^T$ and $X_{k2,p}^T$ are the p-dimensional coordinates of two randomly chosen starfish, whereas A_1 and A_2 represent two random values within the interval (-1, 1), and p is the randomly picked integer within the D dimensions. E_t represents the energy of the starfish. If a starfish's new location is out of bounds, the starfish will not move to the updated location. Instead, it will remain in the previous location (Equation 7) [24].

Exploitation phase:

In SFOA, hunting and regeneration behaviors are simulated in the exploitation phase.

- The hunting model of starfish is also modeled in Equation 10-11.

$$d_m = \left(X_{best}^T - X_{m_p}^T\right), \quad m = 1, 2, \dots, 5 \tag{10}$$

$$Y_i^T = X_i^T + r_1 d_{m1} + r_2 d_{m2} \tag{11}$$

d_m represents the five distances measured between the global best and the other starfish, whereas m_p denotes five randomly selected starfish. r_1 and r_2 represent random values within the range of 0 to 1, while d_{m1} and d_{m2} are chosen randomly from the set d_m [24].

- Starfish are slow-moving and can be caught by predators. If a predator catches a starfish, the starfish may cut off one of its arms. In this case, the regeneration phase occurs in the SFOA. This only applies to the last starfish in the population (i = N). The regeneration phase is modeled in Equation 12 [24].

$$Y_i^T = exp\left(-T \times \frac{N}{T_{max}}\right) X_i^T \tag{12}$$

If the new location obtained by Equation 11 or Equation 12 is outside the search space, the new location is adjusted as shown in Equation 13.

$$X_i^{T+1} = \begin{cases} Y_i^T & l_b \le Y_i^T \le u_b \\ l_b & Y_i^T < l_b \\ u_b & Y_i^T > u_b \end{cases} \tag{13}$$

Algorithm 1 illustrates the pseudo-code of SFOA, while Figure 1 illustrates the flowchart of SFOA. [24].

Algorithm 1: Pseudo-code of SFOA [24]

Start SFOA

Input: Setting the input parameters of SFOA and the problem (N, D, F, T_{max}, u_b, l_b, G_p)

1: Creating the starfish population using Eqs. (1) and (2)
2: Evaluation of the starfish population using Eq. (3)
3: Setting i=1 and T=1
4: **While** $T < T_{max}$ **Do**
5: **If** *rand* > G_p **Do** **% Exploration phase**
6: Calculate θ and E_t by Eq. (6) and Eq. (9), respectively.
7: **For** i=1 **to** N
8: **If** $D > 5$ **Do**
9: Generate five p-dimensions among D
10: **For** j=1 **to** 5
11: Calculate a_1 by Eq. (5)
12: Update dimensions of the positions of the $starfish$ using Eq. (4)
13: **End For**
14: Check the boundary of position by Eq. (7)

15: **Else**
16: Generate two random numbers A_1 and A_2
17: Select a random dimension p
18: Update the p-dimension using Eq. (8)
19: Check the boundary of position by Eq. (7)
20: **End If**
21: **End For**
22: **Else** **% Exploitation phase**
23: Generate distances between the best position and starfish
24: **For** i=1 **to** N
25: Generate r_k and update the position using by Eq. (11)
26: **If** i=N **Do**
27: Update the position using by Eq. (12)
28: **End If**
29: Check the boundary of position by Eq. (13)
30: **End for**
31: **End If**
32: Calculate the fitness values for all starfish
33: Save the best starfish found so far
34: T=T+1
35: **End While**
36: Print the best starfish
End SFOA

2.2. Convolutional Neural Networks (CNNs)

Convolutional neural networks (CNNs) are a deep learning method that is frequently used in computer image recognition studies and yields very successful results. Unlike multilayer ANN (MLP), in CNN, instead of matrix multiplication, convolution (the movement of filters over the input for feature extraction) is used in at least one of its layers. In the first layers, feature extraction is performed on the given input via filters. At the same time, dimensionality reduction functions are employed to reduce the computational cost and to transmit the summary information of the features learned from the input to the other layers. The fully connected layer or layers are then fed these features from the input, which are converted into a one-dimensional vector. The classification process is then conducted [1], [38], [39], [40], [41].

2.2.1. CNNs structure

Input Layer: Once the image being input is transformed into a matrix of numerical values, it may first be sent to the input layer. The size of the matrix may vary based on the image's width and height properties. For example, while a 28×28 grayscale image is expressed with a 28×28×1 matrix; for a color image of the same size, it is expressed with a 28×28×3 matrix due to the RGB (Red, Green, Blue) channels [41].

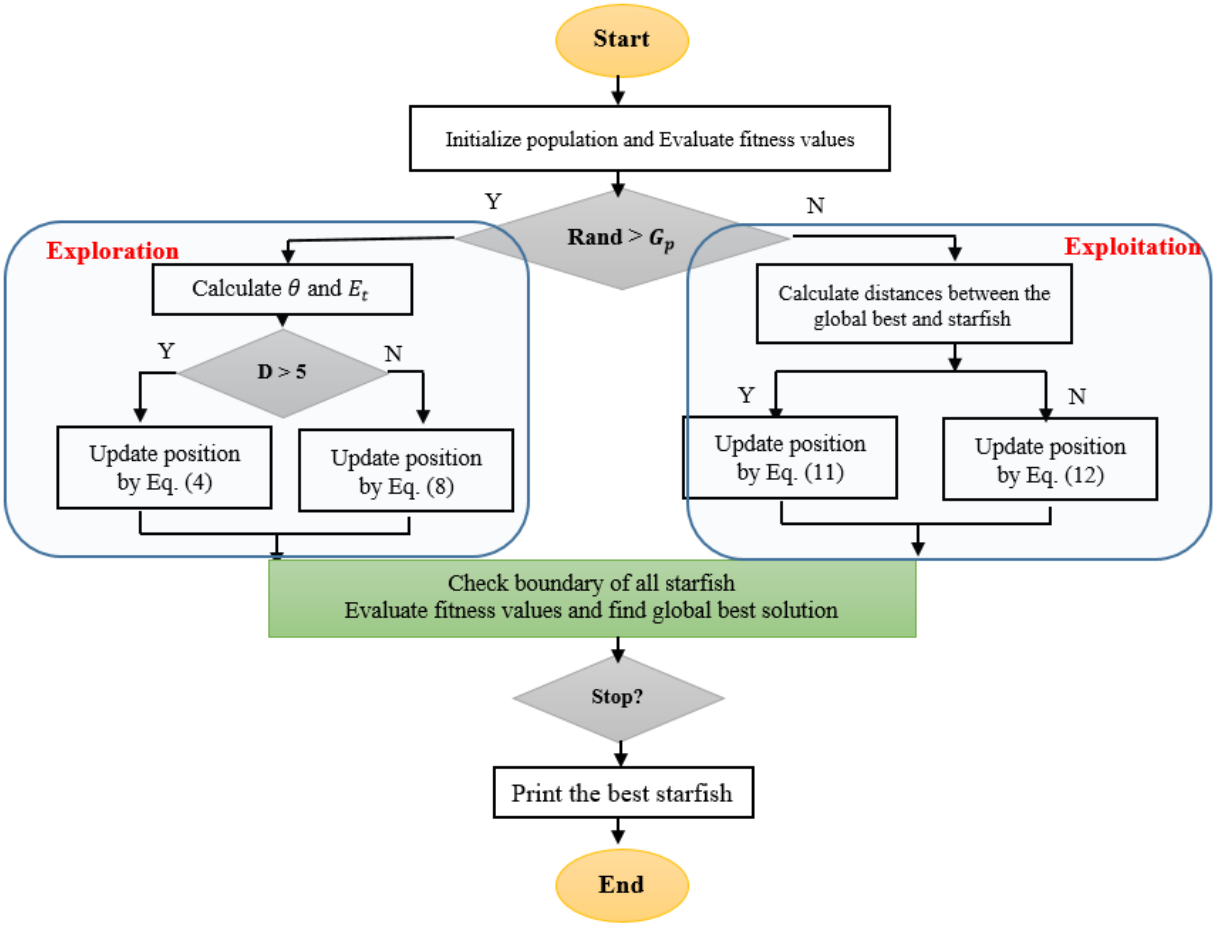


Figure 1. The flowchart of the SFOA [24]

Convolution Layer: New feature sets are attempted to be extracted in this layer by implementing filters that are based on the inputs from the input layer. Matrices of a specific dimension are used to depict each filter that has been applied. Each filter applied on the input tries to reveal a different feature. The selection of the filter sizes is very important. Filters of very large sizes may cause some features not to be detected on the input. The coefficients of the filters represent the weights in the ANNs and new filters are obtained by updating these filter coefficients according to the error value at each step. Each filter on the input traverses all pixels in order. The coefficients of the input and the coefficients of the filter are multiplied pointwise, and this procedure is carried out for each channel of the input. Feature maps are extracted in Equation 14. The input matrix and the weight matrices are multiplied. $X^{[i]}$ is an input matrix of layer i, $W^{[i-1]}$ is a matrix that holds the values of each filter applied in layer i-1, $Z^{[i]}$ is the matrix in which the new feature maps resulting from the processing of the weight and input matrices are stored [41].

$$Z^{[i]} = X^{[i]}W^{[i-1]} + b^{[i-1]} \tag{14}$$

Pooling Layer: The pooling process is used to reduce the size of the data and make the model work more efficiently. During this process, the height and width dimensions of the image are reduced in particular. Although this reduction causes some information loss, it contributes to faster learning of the neural network model. There are no parameters to be learned in the pooling layer; only hyperparameters such as pool size and stride are determined. The most commonly used pooling methods include maximum pooling and average pooling. For example, in the maximum pooling method, the filter navigates over the

input matrix in the specified step range and takes the highest value at its position, reflected in the output matrix [41].

Fully Connected Layer: In convolutional neural networks, the last layers usually consist of fully connected layers that perform classification operations. Until this last layer is reached, intermediate layers apply operations such as feature extraction, dimensionality reduction, and normalization on the data. After the output of the model is obtained, the error is calculated by comparing it with the real value. The weights of the network are rearranged depending on this error. This process continues until the error value drops to a certain level or a predetermined training step is completed [41].

2.3. Dataset Definition

In this study, MNIST and Kuzushiji-MNIST datasets are used to test the hyper-parameters of the CNN model. Hyper-parameters of a CNN model are optimized using these datasets. The MNIST dataset consists of 70000 black and white handwritten digit images with a size of 28 × 28 pixels. These images represent digits between 0 and 9. The MNIST dataset includes different handwriting styles in each digit class. In other words, there are examples of the same digit written differently by different people. This helps handwriting recognition algorithms to learn to recognize different writing styles [10]. The MNIST dataset is taken from the "http://yann.lecun.com/exdb/mnist" web address. MNIST's database of handwritten digits has a training set of 60000 examples and a test set of 10000 examples. A sample part of the MNIST dataset used is shown in Figure 2.

Kuzushiji-MNIST is a dataset that includes the traditional Kuzushiji handwritten characters of Japan. This dataset is comparable to the MNIST (written figures) dataset; however, it employs Japanese handwritten characters. Kuzushiji are traditional, ornate, and arched characters that are employed in Japanese writing. This writing style is a historical aspect of Japanese culture; however, it is not frequently employed in contemporary Japanese writing systems. The Kuzushiji-MNIST dataset is taken from the "https://www.kaggle.com/datasets/anokas/kuzushiji/data" web address. Kuzushiji-MNIST's database of traditional handwritten characters has a training set of 60000 examples and a test set of 10000 examples. A sample part of the Kuzushiji-MNIST dataset used is shown in Figure 3.

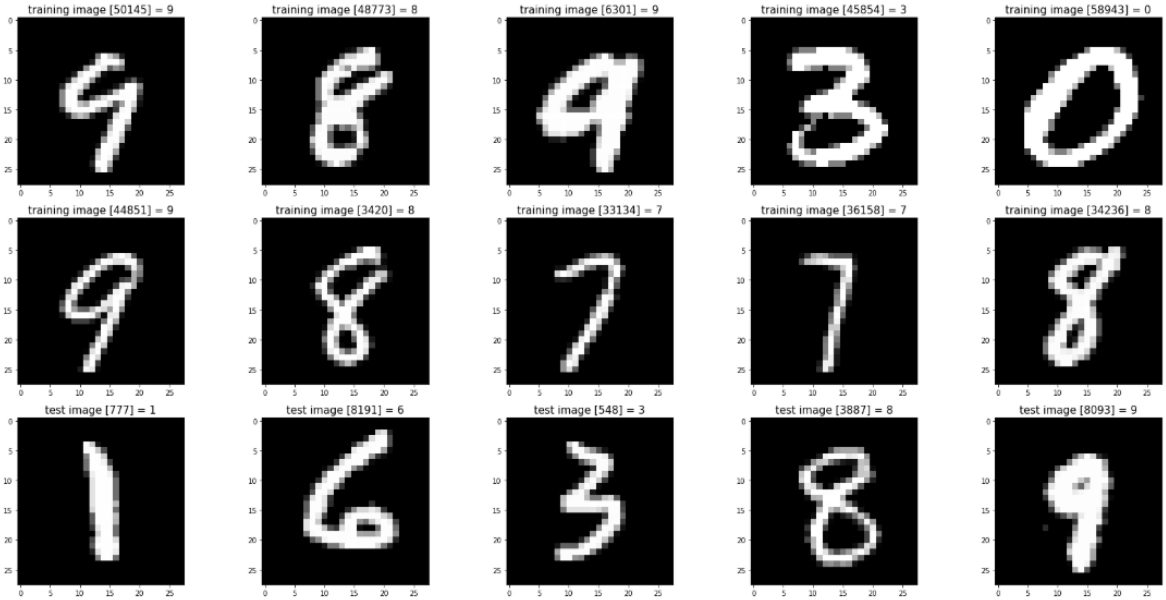


Figure 2. An example from the MNIST dataset (https://www.kaggle.com/code/hojjatk/read-mnist-dataset)

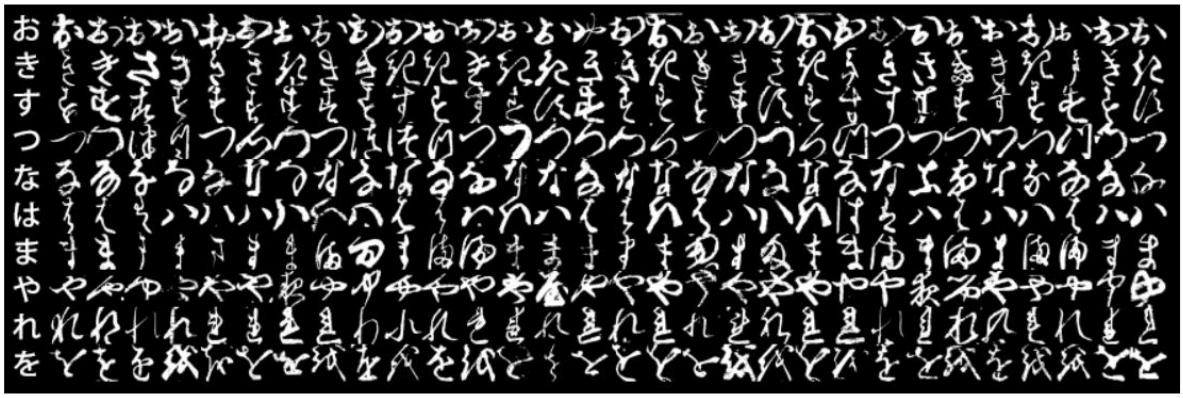


Figure 3. Some examples of Kuzushiji-MNIST dataset (https://www.kaggle.com/datasets/anokas/kuzushiji/data)

2.4. Convolutional Neural Network Architecture and Training Process

This paper presents a convolutional neural network (CNN) architecture for the classification of grayscale pictures of 28×28 pixels. The model consists of two convolutional blocks and three fully connected layers, providing a structure suitable for parameter optimization due to its simplicity. The number of filters in the convolutional layers, the size of the kernels, and other structural details, along with training settings like learning rate, number of epochs, batch size, and the optimization algorithm, were looked at as part of the hyperparameters. Different combinations of these hyperparameters were tested, and their impact on how well the model classified was examined. Different combinations of these hyperparameters were tested, and their effects on the model's classification performance were analyzed.

Each convolutional block includes the convolution process, ReLU activation function, batch normalization, and 2×2 maximum pooling layers. With "same" padding, the output dimensions are preserved, and information loss is minimized. Following these blocks, there are three fully connected layers consisting of 120, 84, and 10 neurons, respectively, with ReLU activation applied after each layer. The last layer generates multi-class output using softmax and classification layers. The model was trained in the MATLAB environment, and the training data was enhanced using various data augmentation techniques. The learning rate was randomly selected within a specific range. At the end of the training process, the model was evaluated for classification performance on the test data.

2.5. Parameter Measurement Metrics

In this study, a CNN model is developed on the MNIST and Kuzushiji-MNIST datasets. Hyperparameter analyses of the CNN model are performed with SFOA, a newly proposed metaheuristic algorithm in recent years. Six different hyperparameters (channel number, kernel size, batch size, learning rate, optimizer type, and epoch size) of the CNN model are analyzed in detail. The role of the determined hyperparameters in the CNN model is listed below.

- *Channel number:* This parameter denotes the number of filters used in each layer.
- *Kernel Size:* The kernel functions as a sliding window across the image, processing the pixels contained inside the window. The kernel size denotes the dimensions of this window in terms of width and height.
- *Batch Size:* The batch size denotes the number of data points the model processes simultaneously during training.
- Learning Rate: The learning rate dictates the number of steps required to update the model's weights.

- *Optimizer Type:* Optimizer Types are methods of updating the weights to minimize the model's loss function.
- *Epoch Size:* An epoch refers to the model processing the whole training dataset one time. The epoch size denotes the number of iterations the model will pass through the dataset during the training process.

The values of the hyperparameters to be optimized are presented in Table 1.

Table 1. Hyper parameters and their selectable values

Hyperparameters	Selectable values
Channel number	8, 16, 32, 64
Kernel Size	1×1, 3×3, 5×5, 7×7, 9×9
Batch Size	32, 64, 128, 256
Learning Rate	$0 \leq l \leq 0.05$
Optimizer Type	sgdm, adam, rmsprop
Epoch Size	5, 10, 15, 20

Upon determining the hyperparameter values to be optimized, candidate solutions are generated randomly. The channel number parameter can assume four different values, as illustrated in Table 1. Random values are generated within the range of 1 to 4. The generated random values are converted into the hyperparameter values they represent. In the situation that the randomly generated value for the channel number parameter equals 2, the channel number parameter will be assigned the value of 16. When the generated value for the optimization algorithm equals 1, the optimization algorithm used is sgdm. Table 2 presents an example representation of candidate solutions, detailing hyperparameters and their associated values.

Table 2. A candidate solution and corresponding hyperparameter values

	Channel Number	Kernel Size	Batch Size	Learning Rate	Optimizer Type	Epoch Size
Candidate Solution	2	3	4	0.01	2	1
Corresponding Value	16	5x5	256	0.01	adam	5

Additionally, the results are evaluated according to some measurement metrics (Accuracy, Precision, Recall, and F1-Score). These performance metrics have been documented in the literature and discussed extensively in various studies [42], [43], [44]. The mathematical expressions of these comparison metrics are as follows:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{15}$$

$$\text{Precision} = \frac{TP}{TP+FP} \tag{16}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{17}$$

$$F1-Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{18}$$

True positive (TP) represents the number of instances that the model predicted as positive and were actually positive. True negative (TN) represents the negative instances that the model correctly classified

as negative. False positive (FP), also known as Type I error, refers to the instances that are actually negative but are incorrectly labeled as positive by the model. False negative (FN), also known as type II error, refers to the instances that are actually positive but are incorrectly predicted as negative by the model. Table 3 presents an overall assessment of the classification performance with the confusion matrix created over these four basic values.

Table 3. Summary of prediction results for classification problems using confusion matrix [42]

Actual Class	**Expected class**	
	Negative	**Positive**
Positive	False negative (*FN*)	True positive (*TP*)
Negative	True negative (*TN*)	False positive (*FP*)

3. RESULTS AND DISCUSSION

In this study, the hyperparameters of the CNN model were determined using the SFOA, and the MNIST and Kuzushiji-MNIST datasets were utilized in this testing process. For SFOA, the population size (*N*) was set at 20, and the runtime value was set at 5. These values were chosen based on existing literature to achieve more reasonable runtime durations [10], [45]. ReLU (Rectified Linear Unit), which returns 0 for negative inputs and retains the same value for positive inputs, was chosen as the activation function. All experimental procedures are conducted on a computer equipped with an Intel(R) Core(TM) i5-1135G7 CPU running at 2.40 GHz, featuring 8.00 GB of RAM, and operating on Windows 10.

The hyperparameter values that achieve the highest accuracy value for the MNIST and Kuzushiji-MNIST datasets are presented in Table 4. According to Table 4, the parameters except for batch size and learning rate are the same for both datasets. In the test conducted on the MNIST dataset, the batch size was selected as 256. If we need to interpret it according to Table 5, it can be said that selecting a high batch size increases the accuracy rate, puts the model into a more stable learning process, and calculates the gradients more accurately. In the test conducted on the Kuzushiji-MNIST dataset, the batch size was selected as 64. The fact that it obtained lower accuracy suggests that more frequent updates and gradients containing more noise may negatively affect the learning process of the model. However, small batch sizes generally provide better generalization, so there is also a possibility that the accuracy difference is specific to the dataset. In this study, the learning rate hyperparameter was set between 0 and 0.05, and different values were obtained for the highest accuracy in both datasets. The learning rate was determined as 0.011 in the MNIST dataset and 0.035 in the second dataset. These results indicate that each dataset has its own learning dynamics, and the optimal learning rate may vary depending on the characteristics of the dataset. The measurement metrics for the MNIST and Kuzushiji-MNIST datasets are presented in Table 5. Accordingly, the MNIST dataset achieved an accuracy of 99.52%, whereas the Kuzushiji-MNIST dataset achieved an accuracy of 97.91%. The model that was developed has attained substantial success, as indicated by these findings. The changing graphics in Figure 4 indicate that accuracy improves with an increasing number of runs in both datasets. This situation establishes an expectation that the accuracy metric achieved by the model across both datasets would improve with an increased number of runs. Figure 5 illustrates the confusion matrices for the MNIST and Kuzushiji-MNIST datasets. The model's overall accuracy is quite high, as evidenced by both confusion matrices, and the majority of classes are correctly classified. Nevertheless, it has been noted that the model's performance may be impacted, and misclassifications may occur as a result of the presence of comparable features among certain classes. The utilization of additional training data and feature engineering (such as feature selection, data augmentation and normalization, etc.) is thought to be effective in minimizing these errors.

Table 4. Hyperparameter values that obtain the highest accuracy value for MNIST and Kuzushiji-MNIST datasets

	Channel number	Kernel Size	Batch Size	Learning Rate	Optimizer Type	Epoch Size
MNIST						
Solution	4	3	4	0.011	1	4
Corresponding Value	64	5x5	256	0.011	sgdm	20
Kuzushiji-MNIST						
Solution	4	3	2	0.035	1	4
Corresponding Value	64	5x5	64	0.035	sgdm	20

Table 5. Measurement metrics for MNIST and Kuzushiji-MNIST datasets

Datasets	Accuracy	Precision	Recall	F1-Score
MNIST	0.9952	0.9982	0.9947	0.9964
Kuzushiji-MNIST	0.9791	0.9751	0.9800	0.9775

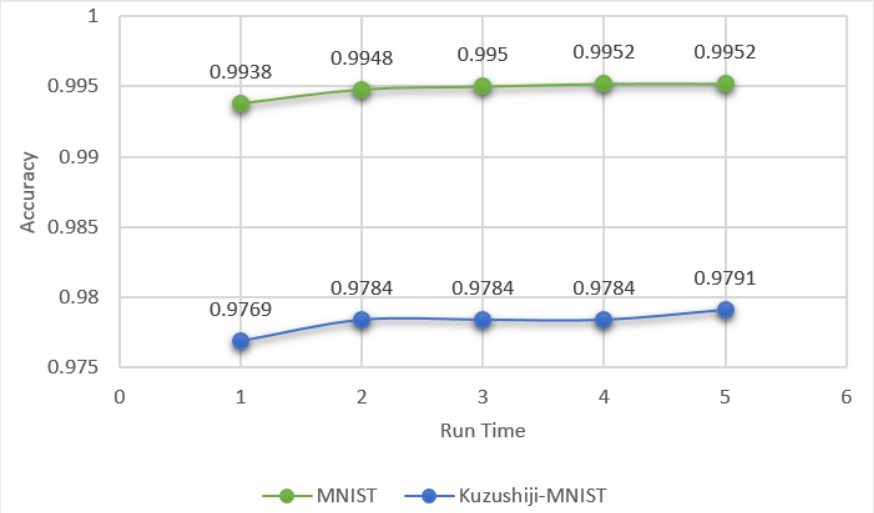
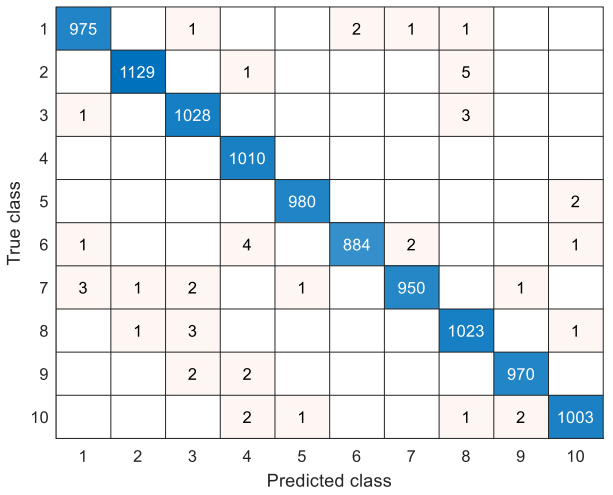
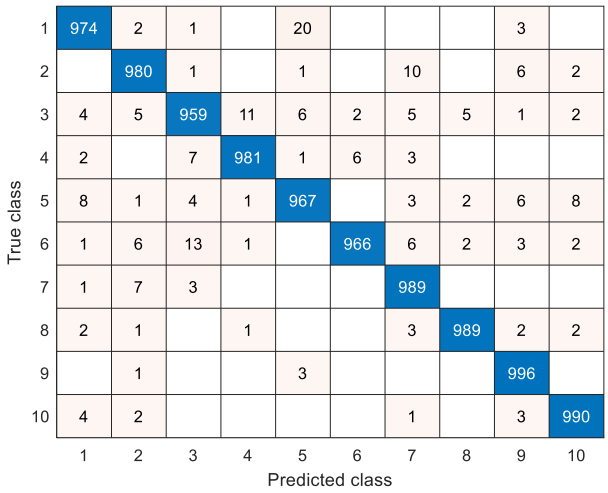


Figure 4. Accuracy changing for MNIST and Kuzushiji-MNIST datasets



(a)



(b)

Figure 5. (a) The confusion matrix for the MNIST dataset, **(b)** Confusion matrix for the Kuzushiji-MNIST dataset

The accuracy values obtained from the proposed model were compared with recent studies in the literature and the comparison results are presented in Table 6. Furthermore, for a fairer comparison, hyperparameter analysis was conducted using metaheuristic algorithms, and the results from Reference [10] are also included in Table 6. Reference [10] also performed hyperparameter analysis of a CNN model on a similar dataset (MNIST) using the Scatter Search Algorithm (SSA). In addition to this study, the Animated Oat Optimization Algorithm (AOO) [46] analyzed similar hyperparameter values under comparable conditions, such as similar population size and maximum iteration, as SFOA. Table 6 also includes the results of AOO. In this way, to ensure a fair comparison of SFOA results, Table 6 presents hyperparameter comparisons with different metaheuristic algorithms (SSA and AOO) on similar datasets (MNIST and Kuzushiji-MNIST).

According to the results in the Table 6, the proposed model achieved superior success by obtaining the highest accuracy value compared to the methods it used in the MNIST dataset. The accuracy value obtained by the proposed model in the Kuzushiji-MNIST dataset was ranked second compared to the methods it was compared to. In light of these results, it has been proven that the proposed method in this study is successful and preferable compared to the literature.

Table 6. Literature comparison results for MNIST and Kuzushiji-MNIST datasets

MNIST	
Studies in literature	**Accuracy (%)**
Proposed model	**99.52**
Hyperparameter optimization+CNN+Animated Oat Optimization Algorithm(AOO) [46]	99.45
Hyperparameter optimization+CNN+Scatter search algorithm+crossover strategy [10]	93.24
CNN+ Alexnet model [47]	98.52
CNN+ Q-learning+ reinforcement learning [48]	98.97
Hyperparameter optimization + CNN [7]	99.40
Hyperparameter optimization + CNN+ artifcial bee colony [49]	99.21
Kuzushiji-MNIST	
Studies in literature	**Accuracy (%)**
Proposed model	97.91
Hyperparameter optimization+CNN+Animated Oat Optimization Algorithm(AOO) [46]	96.47
CNN+ Single-Iteration Optimization [50]	96.33
Simple CNN [51]	96.13
CNN+ residual shrinkage balanced network [52]	89.01
CNN+ self-organized+ without backpropagation training [53]	95.03
CNN+ multi-objective grammatical evolution framework [54]	**98.27**

Lastly, this section evaluates the performance of the proposed model on a different image dataset. The EMNIST Balanced dataset was used for this analysis. EMNIST (Extended MNIST) is a more comprehensive version of MNIST, developed for classifying handwritten letters and numbers. EMNIST Balanced, a subset of this dataset, is designed to ensure class balance and reduce classification errors caused by case differences. The dataset, which contains a total of 47 balanced classes, includes 112800 training examples and 18800 testing examples. Therefore, this dataset, with a total of 131600 samples, is one of the most balanced options in terms of generalization performance. The EMNIST dataset is taken from the "https://www.kaggle.com/datasets/crawford/emnist" web address.

Accordingly, the proposed model was first evaluated on the EMNIST dataset using the same hyperparameter settings as those used for MNIST and Kuzushiji-MNIST. Then, the model was run under similar conditions to find the best hyperparameters for EMNIST. The results of these experiments are shown in Table 7.

When the hyperparameters identified by the proposed model for the MNIST and Kuzushiji-MNIST datasets were applied to the EMNIST dataset, accuracies of 88.65% and 88.73% were achieved, respectively. Subsequently, the proposed model also identified hyperparameters specific to the EMNIST dataset, leading to an accuracy of 88.71%. The results show that the hyperparameters found by the proposed model for different datasets can achieve similar success on more complex datasets, such as EMNIST. The similarity of some parameters, like the number of channels and kernel size, supports this close accuracy. This demonstrates that the proposed model not only finds dataset-specific parameters but also produces hyperparameters that are applicable across different datasets.

Table 7. Comparison of Model Performance on MNIST, Kuzushiji-MNIST, and EMNIST Datasets

	Channel Number	Kernel Size	Batch Size	Learning Rate	Optimizer Type	Epoch Size	EMNIST Accuracy (%)
MNIST Solution	4	3	4	0.011	1	4	88.65
Corresponding Value	64	5x5	256	0.011	sgdm	20	
Kuzushiji-MNIST Solution	4	3	2	0.035	1	4	88.73
Corresponding Value	64	5x5	64	0.035	sgdm	20	
EMNIST Solution	4	3	3	0.042	1	4	88.71
Corresponding Value	64	5x5	128	0.042	sgdm	20	

4. CONCLUSIONS

Convolutional Neural Networks (CNNs) are among the artificial intelligence algorithms that have been applied effectively and intensively in recent years, yielding successful results in many real-world problems. In this study, the parameters used for training CNNs and the layers used in constructing the network structure were optimized to identify the hyperparameters that provide the highest accuracy. The selected hyperparameters included the number of channels, kernel size, batch size, learning rate, optimizer type, and epoch size. The newly proposed Starfish Optimization Algorithm (SFOA) was employed for hyperparameter optimization. When the literature is examined, it is seen that PSO and GA algorithms are generally preferred in hyperparameter optimization for deep network models. Today, there are metaheuristic algorithms that are much more effective than PSO and GA and have the ability to systematically explore and use the search space. SFOA is one of these successful metaheuristic algorithms. For the first time in this study, hyperparameter optimization was performed on CNNs model with SFOA. SFOA is an algorithm inspired by the lifestyle of sea stars. The proposed model was tested on two different datasets (MNIST and Kuzushiji-MNIST), achieving accuracy rates of 99.52% and 97.91% for MNIST and Kuzushiji-MNIST, respectively, as a result of hyperparameter optimization. When comparing these obtained results with those in the literature, it has been demonstrated that the proposed model is highly successful. In addition, the similar hyperparameters and accuracy values obtained by the proposed model for different datasets showed that the model can produce generalizable and transferable hyperparameters.

In future studies, the proposed model can be evaluated on more diverse datasets, and its performance can be further investigated. Additionally, research can focus on enhancing performance through modifications to the SFOA.

Declaration of Ethical Standards

No conflict of interest or common interest has been declared by the authors.

Credit Authorship Contribution Statement

The authors contributed equally to the study.

Declaration of Competing Interest

There is no competition or conflict of interest among the authors.

Funding / Acknowledgements

The author (s) has not received any financial support for the research, authorship, or publication of this study.

Data Availability

- *SFOA code:* Zhong Changting (2025). Starfish Optimization Algorithm (SFOA) (https://www.mathworks.com/matlabcentral/fileexchange/173735-starfish-optimization-algorithm-sfoa), MATLAB Central File Exchange. Retrieved April 15, 2025.
- The MNIST dataset for the CNNs referenced in this study can be accessed at https://www.kaggle.com/code/hojjatk/read-mnist-dataset and http://yann.lecun.com/exdb/mnist.
- The Kuzushiji-MNIST dataset for the CNNs referenced in this study can be accessed at https://www.kaggle.com/datasets/anokas/kuzushiji/data.
- The EMNIST dataset for the CNNs referenced in this study can be accessed at https://www.kaggle.com/datasets/crawford/emnist.
- We would like to express our gratitude to these sources for providing valuable data for our research.

5. REFERENCES

[1] A. Gülcü, Z. Kuş, "Konvolüsyonel sinir ağlarında hiper-parametre optimizasyonu yöntemlerinin incelenmesi," *Gazi University Journal of Science Part C: Design and Technology*, 7(2), 503-522, 2019.

[2] E. Öztemel, *Yapay Sinir Ağları*. Istanbul: Papatya Yayincilik, 2003.

[3] J. Bergstra, R. Bardenet, Y. Bengio, B. Kégl, (2011). "Algorithms for hyper-parameter optimization," in Proceedings of the 24th International Conference on Neural Information Processing Systems NIPS 2011, 2546–2554.

[4] M. Lindauer, K. Eggensperger, M. Feurer, A. Biedenkapp, D. Deng, C. Benjamins, T. Ruhkopf, R. Sass, and F. Hutter, "SMAC3: A Versatile Bayesian Optimization Package for Hyperparameter Optimization," *The Journal of Machine Learning Research.*, 23(1), 2022.

[5] L. Li, K. Jamieson, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization," *Journal of Machine Learning Research*, 18(1), 6765–6816, 2017.

[6] O.M. Khanday, S. Dadvandipour, M.A. Lone, "Effect of filter sizes on image classification in CNN: A case study on CFIR10 and fashion-MNIST datasets," *IAES International Journal of Artificial Intelligence*, 10 (4), 872–878, 2021.

[7] H. Shao, E. Ma, M. Zhu, X. Deng, S. Zhai, "MNIST Handwritten Digit Classification Based on Convolutional Neural Network with Hyperparameter Optimization," *Intelligent Automation & Soft Computing*, 36(3), 2023.

[8] B. Bischl, M. Binder, M. Lang, T. Pielok, J. Richter, S. Coors, J. Thomas, T. Ullmann, M. Becker, A. Boulesteix, D. Deng, M. Lindauer, "Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 13, 2023.

[9] T. Yigit, S. Atmaca, R. Gurfidan, R. Colak, " Hyper Parameter Analysis in Recognition of Handwritten Digits Using Convolutional Neural Network," *Gazi Journal of Engineering Sciences*, 9(4), 268-277, 2017

[10] H. Orhan, D. S. Polat, H. Hakli, H. "Farkli Çaprazlama Teknikleri Kullanan Dağinik Arama Algoritmasi Ile Evrişimli Sinir Ağlarinda Hiper Parametre Optimizasyonu," *Kahramanmaraş Sütçü İmam Üniversitesi Mühendislik Bilimleri Dergisi*, 27(4), 1437-1450, 2024.

[11] Xiao, X., Yan, M., Basodi, S., Ji, C., Pan, Y. (2020). Efficient hyperparameter optimization in deep learning using a variable length genetic algorithm. *arXiv preprint arXiv:2006.12703*.

[12] E. Tanyıldızı, F. Demirtaş, (2019). "Hiper Parametre Optimizasyonu Hyper Parameter Optimization," Paper presented at the 2019 1st International Informatics and Software Engineering Conference (UBMYK).

[13] E. Özbay, F. A. Özbay, "Parçacık Sürüsü Optimizasyon Algoritması ile Optimize Edilmiş Evrişimsel Sinir Ağı Kullanılarak Dermoskopik Görüntülerden Cilt Kanserinin Sınıflandırılması," *Fırat Üniversitesi Mühendislik Bilimleri Dergisi*, 35(1), 261-273, 2023.

[14] Ö. İnik, "CNN hyper-parameter optimization for environmental sound classification," *Applied Acoustics*, 202, 109168, 2023.

[15] M.E. Kıymaç, (2022). "Hyper-parameter optimization of deep neural networks with metaheuristic algorithms," *Yüksek Lisans Tezi*, Alparslan Türkeş Bilim ve Teknoloji Üniversitesi.

[16] M. Yurdakul, (2022). "Meta-sezgisel algoritmalar ile konvolüsyonel sinir ağı mimarisinin hiper parametrelerinin optimizasyonu," *Yüksek Lisans Tezi*, Selçuk Üniversitesi.

[17] G. Atteia, N. Abdel Samee, E. S. M. El-Kenawy, A. Ibrahim, "CNN-hyperparameter optimization for diabetic maculopathy diagnosis in optical coherence tomography and fundus retinography," *Mathematics*, 10(18), 3274, 2022.

[18] M. Ait Amou, K. Xia, S. Kamhi, M. Mouhafid, "A novel MRI diagnosis method for brain tumor classification based on CNN and Bayesian Optimization," *In Healthcare*, 10(3), 494, 2022.

[19] H. ZainEldin, S. A. Gamel, E. S. M. El-Kenawy, A. H. Alharbi, D. S. Khafaga, A. Ibrahim, F. M. Talaat, "Brain tumor detection and classification using deep learning and sine-cosine fitness grey wolf optimization," *Bioengineering*, 10(1), 18, 2022.

[20] M. Sethi, S. Ahuja, S. Rani, P. Bawa, A. Zaguia, A. "Classification of Alzheimer's Disease Using Gaussian-Based Bayesian Parameter Optimization for Deep Convolutional LSTM Network," *Computational and Mathematical Methods in Medicine*, 2021(1), 4186666, 2021.

[21] M. Zhang, H. Li, S. Pan, J. Lyu, S. Ling, S., Su, "Convolutional neural networks-based lung nodule classification: A surrogate-assisted evolutionary algorithm for hyperparameter optimization," *IEEE Transactions on Evolutionary Computation*, 25(5), 869-882, 2021.

[22] K. Shankar, Y. Zhang, Y., Liu, L. Wu, C. H. Chen, "Hyperparameter tuning deep learning for diabetic retinopathy fundus image classification," *IEEE Access*, 8, 118164-118173, 2020.

[23] L. M. Seng, B. Bang Chen Chiang, Z. Arabee Abdul Salam, G. Yih Tan, H. Tong Chai, "MNIST handwritten digit recognition with different CNN architectures," *Journal of Applied Technology and Innovation*, 5(1), 7-10, 2021.

[24] C. Zhong, G. Li, Z. Meng, H. Li, A. R. Yildiz, S. Mirjalili, S. "Starfish optimization algorithm (SFOA): a bio-inspired metaheuristic algorithm for global optimization compared with 100 optimizers," *Neural Computing and Applications*, 37(5), 3641-3683, 2025.

[25] D. Izci, S. Ekinci, M. Jabari, M. Bajaj, V. Blazek, L. Prokop, ... & S. Mirjalili, "A new intelligent control strategy for CSTH temperature regulation based on the starfish optimization algorithm," *Scientific Reports*, 15(1), 12327, 2025.

[26] S. S. Kadam, A. C. Adamuthe, A. B. Patil, "CNN model for image classification on MNIST and fashion-MNIST dataset," *Journal of scientific research*, 64(2), 374-384, 2020.

[27] S. Ahlawat, A. Choudhary, A. Nayyar, S. Singh, & B. Yoon, "Improved handwritten digit recognition using convolutional neural networks (CNN)," *Sensors*, 20(12), 3344, 2020.

[28] A. Jain & B. K. Sharma, "Analysis of activation functions for Convolutional Neural Network based mnist handwritten character recognition," *International journal of advanced studies of scientific research*, 3(9), 2018.

[29] A. A. Yahya, J. Tan, & M. Hu, "A novel handwritten digit classification system based on convolutional neural network approach," *Sensors*, 21(18), 6273, 2021.

[30] G. Gilanie, H. Shafiq, S. N. Batool, S. N. Abbas, H. Shafique, S. Cheema, ... & M. Ahsan, "PARAMETER OPTIMIZATION OF AUTOENCODER FOR IMAGE CLASSIFICATION USING GENETIC ALGORITHM," *Spectrum of Engineering Sciences*, 3(4), 201-213, 2025.

[31] S. Kiliçarslan, S. "PSO+ GWO: a hybrid particle swarm optimization and Grey Wolf optimization based Algorithm for fine-tuning hyper-parameters of convolutional neural networks for Cardiovascular Disease Detection," *Journal of Ambient Intelligence and Humanized Computing*, 14(1), 87-97, 2023.

[32] B.S. Park, S. Lee, Y.H. Seo, "Training Method for Enhancing Classification Accuracy of Kuzushiji-MNIST/49 using Deep Learning based on CNN," *Journal of the Korea Institute of Information and Communication Engineering*, 24(3), 355-363, 2020.

[33] S. Zhao, Q. Fan, Q. Dong, Z. Xing, X. Yang, X. He, "Efficient construction and convergence analysis of sparse convolutional neural networks," *Neurocomputing*, 597, 128032, 2024.

[34] B. T. Kettle and J. S. Lucas, "Biometric relationships between organ indices, fecundity, oxygen consumption and body size in Acanthaster planci (L.) (Echinodermata; Asteroidea)," *Bulletin of Marine Science*, 41(2), 541–551, 1987.

[35] P. O. Ottesen and J. S. Lucas, "Divide or broadcast: interrelation of asexual and sexual reproduction in a population of the fissiparous hermaphroditic seastar Nepanthia belcheri (Asteroidea: Asterinidae)," *Marine Biology*, 69, 223–233, 1982.

[36] M. Güler and A. Lök, "Foragig behaviors of sea stars, Marthasterias glacialis and Astropecten aranciacus (Asteroidea) and predator-prey interactions with warty venus clam, Venus verrucose (Bivalvia)," *Journal of Experimental Marine Biology and Ecology*, 465, 99–106, 2015.

[37] Y. Hayashi and T. Motokawa, "Effects of lonic environment on viscosity of catch connective tissue in Holothurian body wall," *Journal of experimental biology*, *125*(1), 71–84, 1986.

[38] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, L. D. Jackel, "Handwritten digit recognition with a back-propagation network," *Advances in neural information processing systems*, 2, 1990.

[39] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, 86 (11), 2278-2324, 1998.

[40] I. Goodfellow, Y. Bengio, A. Courville, Y. Bengio. *Deep learning* (Vol. 1). Cambridge: MIT press, 2016.

[41] M. Tan, C. Emeksiz, "Hydrogen fuel cell parameter estimation using an innovative hybrid estimation model based on deep learning and probability pooling," *International Journal of Hydrogen Energy*, 110, 445-456, 2024.

[42] M. Braik, "Enhanced ali baba and the forty thieves algorithm for feature selection," *Neural Computing and Applications*, 35(8), 6153–6184, 2023.

[43] D. Albashish, A. I. Hammouri, M. Braik, J. Atwan, S. Sah ran, "Binary biogeography-based optimization based SVM-RFE for feature selection," *Applied Soft Computing*, 101, 107026–107045, 2021.

[44] E. A. Mohamed, M. S. Braik, M. A. Al-Betar, M. A. Awadallah, "Boosted spider wasp optimizer for high-dimensional feature selection," *Journal of Bionic Engineering*, 21(5), 2424-2459, 2024.

[45] J. C. Geng, Z. Cui, X. S. Gu, "Scatter search based particle swarm optimization algorithm for earliness/tardiness flowshop scheduling with uncertainty," *International Journal of Automation and Computing*, 13(3), 285-295, 2016.

[46] R. B. Wang, R. B. Hu, F. D. Geng, L. Xu, S. C. Chu, J. S. Pan, ... & S. Mirjalili, "The Animated Oat Optimization Algorithm: A nature-inspired metaheuristic for engineering optimization and a case study on Wireless Sensor Networks," *Knowledge-Based Systems*, 113589, 2025.

[47] A. Kumar, I. Singh, B. Singh. *Classification of MNIST Dataset Using Different CNN Architecture. In Intelligent Circuits and Systems for SDG 3–Good Health and well-being* (pp. 273-280). CRC Press, 2024.

[48] F. M. Talaat and S.A. Gamel, "RL based hyper-parameters optimization algorithm (ROA) for convolutional neural network," *Journal of Ambient Intelligence and Humanized Computing*, 14(10), 13349-13359, 2023.

[49] U. Erkan, A. Toktas, D. Ustun, "Hyperparameter optimization of deep CNN classifier for plant species identification using artificial bee colony algorithm," *Journal of Ambient Intelligence and Humanized Computing*, 14(7), 8827-8838, 2023.

[50] L. C. Ribeiro, G. H. D. Rosa, D. Rodrigues, J. P. Papa, "Convolutional neural networks ensembles through single-iteration optimization," *Soft Computing*, 26(8), 3871-3882, 2022.

[51] A. Ghosh, A., Mukherjee, C. Ghosh, (2020). "Simplistic deep learning for Japanese handwritten digit recognition," In Intelligent Techniques and Applications in Science and Technology: Proceedings of the First International Conference on Innovations in Modern Science and Technology 1, (pp. 87-93). Springer International Publishing.

[52] H. Qiu, and J. Dong, "A robust residual shrinkage balanced network for image recognition from Japanese historical documents," *Journal of Sensors*, 2023(1), 8316638, 2023.

[53] T. Erkoç, M. T. Eskıl, "A novel similarity based unsupervised technique for training convolutional filters," *IEEE Access*, 11, 49393-49408, 2023.

[54] C. A. da Silva, D. C. Rosa, P. B. Miranda, F. R. Cordeiro, T. Si, A. C. Nascimento, ... & P. S de Mattos Neto, "A novel multi-objective grammar-based framework for the generation of convolutional neural networks," *Expert Systems With Applications*, 212, 118670, 2023.