Asia-Pacific Journal of Information Technology and Multimedia 15(1) 2026: 262 – 281
https://doi.org/10.17576/apjitm-2026-1501-16

A Novel Approach for Feature Optimization in Deep Learning Models

Pendekatan Baharu bagi Pengoptimuman Ciri dalam Model Pembelajaran Mendalam

Chathurangi Kumari Atugoda[*]*, Subha Fernando*

Department of Computational Mathematics,Faculty of Information Technology, University of Moratuwa,Katubedda, Colombo, SriLanka

[*]*Corresponding author: chathurangiatugoda4@gmail.com*

Received 2 May 2025
Accepted 11 February 2026, Available online 30 June 2026

ABSTRACT

Deep Neural Networks (DNNs) are widely used in image recognition and classification tasks, where performance strongly depends on effective optimization. Conventional gradient-based optimizers such as Stochastic Gradient Descent (SGD), Adam, and AdaDelta are effective, but they may suffer from slow convergence, sensitivity to initialization, and the risk of getting trapped in local minima, particularly in deep architectures. To address these limitations, this study proposes an Enhanced Particle Swarm Optimization integrated Convolutional Neural Network (EPSO-CNN). The novelty of the proposed method lies in embedding a PSO-operated layer after the group convolution stage to optimize intermediate feature maps before classification. The proposed model was evaluated on MNIST, CIFAR-10, and CIFAR-100 datasets and compared with a baseline LeNet-5 model trained using SGD, Adam, and AdaDelta. Experimental results show that the proposed EPSO-CNN achieves improved classification accuracy, faster convergence, and better training stability, especially on more complex datasets. These findings demonstrate that PSO-based feature map optimization can serve as an effective complementary strategy to conventional gradient-based learning in CNNs.

Keywords**:** Deep neural network; Feature optimization; Particle swarm optimization; Swarm intelligence; Weight values

ABSTRAK

Rangkaian Neural Mendalam (DNN) digunakan secara meluas dalam tugasan pengecaman dan pengelasan imej, yang mana prestasinya sangat bergantung pada pengoptimuman yang berkesan. Pengoptimium berasaskan kecerunan konvensional seperti Penurunan Kecerunan Stokastik (SGD), Adam, dan AdaDelta adalah berkesan, namun ia boleh mengalami penumpuan yang perlahan, kepekaan terhadap pemulaan, dan risiko terperangkap dalam minimum setempat, terutamanya dalam seni bina yang mendalam. Untuk menangani had ini, kajian ini mencadangkan Rangkaian Neural Konvolusi bersepadu Pengoptimuman Kumpulan Zarah Dipertingkat (EPSO-CNN). Kebaharuan kaedah yang dicadangkan ini terletak pada

penyebatan lapisan kendalian PSO selepas peringkat konvolusi kumpulan untuk mengoptimumkan peta ciri perantaraan sebelum pengelasan. Model yang dicadangkan telah dinilai pada set data MNIST, CIFAR-10, dan CIFAR-100 serta dibandingkan dengan model garis dasar LeNet-5 yang dilatih menggunakan SGD, Adam, dan AdaDelta. Hasil eksperimen menunjukkan bahawa EPSO-CNN yang dicadangkan mencapai ketepatan pengelasan yang lebih baik, penumpuan yang lebih pantas, dan kestabilan latihan yang lebih baik, terutamanya pada set data yang lebih kompleks. Penemuan ini membuktikan bahawa pengoptimuman peta ciri berasaskan PSO boleh berfungsi sebagai strategi pelengkap yang berkesan kepada pembelajaran berasaskan kecerunan konvensional dalam CNN.

Kata Kunci: Rangkaian neural mendalam; Pengoptimuman ciri; Pengoptimuman kumpulan zarah; Kecerdasan kumpulan; Nilai pemberat

INTRODUCTION

In recent years, Deep Neural Networks (DNNs) have gained widespread importance across a wide range of fields, including engineering, science, bio-medical science, and robotics. These networks have become essential tools for numerous applications such as classification, regression, pattern recognition, structured prediction, anomaly detection, and decision making, due to their powerful capabilities in solving nonlinear problems. Among these networks, Deep Convolutional Neural Networks (DCNNs) are particularly noteworthy.
DCNNs consist of two main parts: a feature extraction component and a classification component, and they are extensively utilized for image classification tasks. Given their complexity and the critical nature of their applications, optimization is an essential aspect of neural network design and training. Optimization techniques are essential in Engineering and Science, particularly during the design phase to derive the optimal solution from numerous viable alternatives. The primary goal of optimization in neural networks is to improve their performance by fine-tuning various features such as weight values, hyperparameters, and network architecture. Despite the broad scope of optimization, previous research has predominantly focused on optimizing the weight values of neural networks.

Traditional optimization approaches such as SGD, Adam, AdaDelta, and AdaGrad, although widely used, exhibit specific weaknesses when applied to deep CNNs on large-scale image datasets. These methods rely heavily on gradient information, making them highly sensitive to noisy updates and poor weight initialization. In complex networks, this often leads to unstable training dynamics, prolonged convergence, and a tendency to converge to sharp local minima that generalize poorly. Furthermore, their performance depends heavily on careful hyperparameter tuning, and suboptimal settings can significantly degrade classification accuracy. These limitations highlight the need for complementary optimization strategies that can perform global search, are less sensitive to initialization, and reduce reliance on extensive manual tuning.

Thus, there is an increasing need for modern optimization techniques to address these limitations. Swarm Intelligence (SI) based Metaheuristic Optimization techniques have emerged as a viable approach in this context. Swarm Intelligence refers to the collective problem-solving capability that emerges from interactions among individuals and their environment in the search for potential solutions. The fundamentals and advancements in SI algorithms for solving various real-life optimization problems are well-documented in numerous books and research papers. One notable application of SI algorithms is in Artificial

Neural Network Training. Several studies have demonstrated the effectiveness of different SI algorithms in optimizing neural networks for various applications. Each individual in an SI algorithm engages in a set of activities that may involve random searches, feedback mechanisms, and interactions with other individuals. Despite their simplicity, these processes can yield remarkable results. Most SI approaches incorporate both exploration (diversification) and exploitation (intensification) into their functioning. Many prominent algorithms used in past studies integrate these principles. In recent years, novel algorithms such as the Grey Wolf Optimizer (GWO), Grasshopper Optimizer (GO), Firefly Algorithm (FFA), Fireworks Algorithm (FWA), and Bat Algorithm (BA) have also been proposed. The proliferation and popularity of these algorithms can be attributed to their simplicity of inspiration, adaptability, and derivative-free nature. Previous studies have shown that applying SI algorithms to optimize neural networks enhances performance in various fields, including medical image analysis, plant disease identification, and hydrological information prediction. Among these SI approaches, Particle Swarm Optimization (PSO) is widely regarded as an effective method because of its simplicity, relatively small number of control parameters, and derivative-free nature (Ahmadzadeh, 2017).

PARTICLE SWARM OPTIMIZATION

Inspired by the social behavior of bird flocks and fish schools, PSO is a novel stochastic optimization approach (Kennedy and Eberhard, 1995). PSO has since become one of the most prevalent SI algorithms for training neural networks, alongside ACO and ABC (Bui et al.,2018). The principle behind PSO is based on a leader guiding the group, with others following, emulating the social behavior observed in nature. In this context, a feasible solution is referred to as a particle. Particles adjust their positions and velocities to find the best solution based on their individual best prior locations and the swarm's global best solution.
The velocity and position of a particle in 'd' dimensions are described as follows (Chhabra et al.,2018)

Velocity update equation:

$$V_{id}(t+1) = \omega.V_{id}(t) + C_p.r_p.\left(pbest_{id} - X_{id}(t)\right) + C_g.r_g.\left(gbest_{id} - X_{id}(t)\right) \quad (1)$$

Position update equation:

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1) \quad (2)$$

Where:
V_{id} = velocity of the ith particle,
X_{id} = position of the ith particle,
ω=inertia weight,
C_p, $C_{g.}$ = acceleration coefficients,
r_p, r_g = random integers within [0,1],
$pbest_{id}$ = The best local solution,
$gbest_{id}$ = The best global solution

Artificial Neural Networks (ANNs) are effective in categorizing only a handful of classes. However, problems arise when applications require classifying datasets into 100 or more classes, necessitating deeper architectures like DNNs. Conventional optimization approaches such as SGD, Adam, and AdaDelta have outstanding outcomes in ANNs but poorly in DNNs due to the increased complexity and numerous parameters that need adjustment (Chhachhiya, et al. 2017). In such cases, higher predictive techniques, such as the PSO algorithm, are required to train DNNs. Despite its strengths, using PSO to train neural networks presents challenges, including time consumption, GPU dependency, and difficulties in classifying large

datasets or handling small samples for high-dimensional problems. Therefore, improving DNN performance for extensive datasets with improved SI approaches is essential.

This study aims to improve the performance of DCNNs for image classification on large-scale datasets by employing an Enhanced Particle Swarm Optimization (EPSO)approach to optimize the network more effectively. By refining the PSO algorithm and tailoring it to the specific needs of DCNNs, the method seeks to achieve improved convergence speed, accuracy and overall computational efficiency. Building on earlier PSO-CNN approaches, the proposed method embeds a PSO-operated layer after the group convolution stage, enabling direct optimization of intermediate feature maps before classification. The proposed approach is evaluated on MNIST, CIFAR-10, and CIFAR-100 and compared with a baseline CNN trained using SGD to assess classification accuracy and computational efficiency.

To strengthen the scientific framing of this study, the following research questions are addressed: (1) Does the proposed EPSO-operated layer improve classification accuracy compared with conventional gradient-based optimization methods? (2) Does the integration of the PSO-operated layer enhance convergence speed and training stability in CNN architectures? (3) How does the proposed EPSO-CNN perform across datasets of different complexity, namely MNIST, CIFAR-10, and CIFAR-100? Based on these questions, it is hypothesized that the proposed EPSO-CNN will achieve higher classification accuracy, faster convergence, and better training stability than baseline CNN models trained with conventional optimizers.

RELATED WORK

In recent years, PSO has emerged as a prominent technique in neural network training, demonstrating its potential to optimize weights and biases while addressing challenges such as local minima and convergence efficiency. Khalifa introduced a hybrid approach that applied PSO to optimize the final layer weights of a seven-layer Convolutional Neural Network (CNN), achieving improved classification accuracy on the MNIST dataset while employing gradient descent for earlier layers (Khalifa et al.2017). Similarly, Yoshika Chabbra extended PSO optimization across the entire network, demonstrating faster convergence and reduced training epochs on MNIST (Yoshika Chabbra.2017). Tatsuki advanced this work by proposing a Linearly Decreasing Weight PSO (LDWPSO) algorithm to optimize CNN hyperparameters, achieving notable accuracy improvements on MNIST and CIFAR-10 datasets despite significant computational costs (Tatsuki ,2020). Ahamadzadh demonstrated the utility of PSO as an alternative to gradient-based methods in training Artificial Neural Networks (ANNs), optimizing weights and biases to enhance prediction accuracy in regression tasks (Ahamadzadh,2017). From a practical standpoint, the integration of PSO into neural network applications has yielded promising results. For instance, (Dieu T et.al. 2017) combined PSO with Multi-Layer Perceptron (MLP) Neural Networks to predict soil compression coefficients, outperforming traditional methods but facing scalability challenges with small datasets. Sudarshan and Nitila employed a hybrid PSO and Firefly Algorithm for training Feedforward Neural Networks (FFNNs), achieving high accuracy while encountering limitations in handling high-dimensional data and training large-scale networks (Sudarshan and Nitila 2017). Similarly, Revathi presented a hybrid MSPSO-ImFFA model for medical image classification, optimizing neural network weights to enhance lung cancer detection (Revathi et al. 2020). However, challenges related to the computational demands of large-scale networks and high-dimensional medical imaging datasets hindered scalability and practical applicability in Tatsuki. Building upon these advancements and limitations, this study seeks to enhance the efficiency of Deep Convolutional Neural Networks (DCNNs) for image classification tasks involving large datasets. While related works have demonstrated the potential of PSO to

optimize weights and biases and address issues such as local minima and convergence efficiency, limitations in scalability, computational costs, and handling of high-dimensional data remain significant (Wang et.al. 2020).

Over time, a number of modified PSO variants have been developed to improve convergence behavior and search efficiency. These include strategies such as adaptive inertia weights, velocity clamping, constriction factors, and multi-swarm approaches, as well as more advanced extensions like quantum behaved PSO and hybrid methods that combine PSO with gradient descent. Collectively, these modifications aim to balance exploration and exploitation, stabilize particle dynamics, and reduce premature stagnation. While such improvements have enhanced PSO performance in general optimization tasks and in neural network training, most studies stop at the parameter level. Far less attention has been given to adapting PSO as an internal component of CNNs to optimize feature maps.

In addition to, modified PSO variants, hybrid optimization methods have also been investigated in neural network training. Nandy (2020) reported that hybrid optimization can improve convergence and training effectiveness, while Revathi (2020) showed that hybrid swarm-based approaches can enhance classification performance in neural network applications. These studies demonstrate the value of combining complementary optimization strategies. However, most existing studies focus on parameter optimization rather than direct feature map refinement.

This study aims to address these gaps by adapting PSO for deeper integration within DCNNs. Rather than proposing an entirely new variant, the contribution lies embedding a PSO-operated layer after the group convolution stage, enabling swarm optimization to refine intermediate feature maps. This approach builds on insights from existing PSO modifications, such as adaptive inertia, velocity control, and hybrid strategies while shifting the focus from parameter tuning to representation optimization. This objective is to enhance classification accuracy and training stability on large-scale datasets, while mitigating some of the computational and convergence challenges that limit conventional optimization methods.

METHODOLOGY

This study proposes a methodology for enhancing neural network performance through the integration of a PSO-operated layer within the network architecture. By embedding the PSO-operated layer, the aim is to leverage the inherent strengths of PSO, such as its ability to efficiently explore the solution space and avoid local minima, thereby addressing some of the limitations associated with traditional gradient optimization methods. The primary objective of this research is to evaluate the impact of the PSO-integrated layer on the overall performance of the neural network, with a focus on its ability to improve optimization efficiency and enhance model accuracy across diverse applications. This approach offers a promising direction for advancing optimization techniques in neural network training, potentially contributing to more robust and effective deep learning models.

NEURAL NETWORK ARCHITECTURE WITH EMBEDDED PSO LAYER

The proposed neural network architecture incorporates a PSO operated layer to enhance performance and improve training efficiency. The architecture consists of several key components, including convolution operations, group convolution, a PSO-based layer, and

fully connected layers. The diagram in Figure 1 illustrates the block diagram of the neural network architecture that has been proposed.

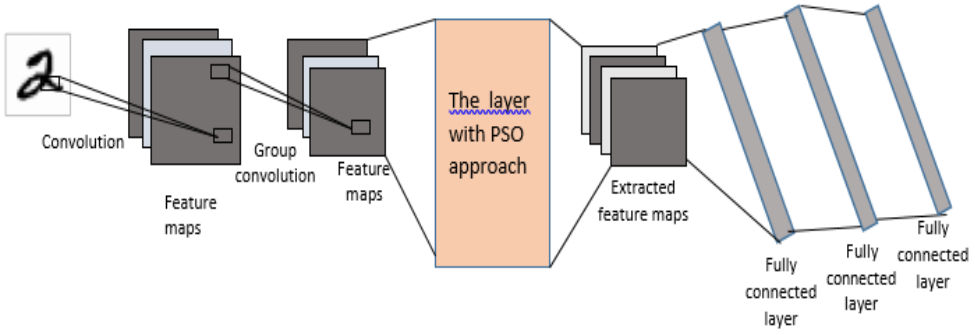

FIGURE 1.Proposed Neural Network Architecture

PSO-OPERATED LAYER

In this study, following the group convolution step, we obtain eight feature maps, each with dimensions of 24x24. As the next phase, we incorporate a layer that utilizes PSO operations. This layer functions as an internal feature extraction layer. Within this framework, each of the eight feature maps is treated as a swarm, and PSO operations are applied separately to each swarm simultaneously. In this context, each particle is represented by a pixel within the feature map, resulting in 576 particles (24 × 24) per feature map. The positions of these particles correspond to the spatial coordinates of the feature map. Initially, random velocities are assigned to each particle. Constants C1 and C2, as well as random numbers r1 and r2, are provided based on values from previous studies. The inertial weight ω is defined as a function of the iterations, starting from a high value and gradually decreasing to enhance local search capabilities. The fitness value for each particle is determined using a categorical loss function. Thus, the fitness value is calculated for each pixel in the feature map, with the pixel yielding the lowest loss designated as the global best (gbest) particle.

PIXEL-LEVEL FITNESS EVALUATION USING CROSS-ENTROPY LOSS

In this study, each pixel in a feature map is modeled as a particle within the PSO framework. The objective is to iteratively adjust the spatial coordinate of these pixels such that the final optimized feature map emphasizes the most discriminative regions for classification.
Let $A \in \mathbb{R}^{H \times W}$ represent the activation feature map, where H and W are its height and width, respectively. Each pixel at coordinate $({x_i}^t, {y_i}^t)$ is considered as a particle ${X_i}^t$ in the swarm at iteration t, with a corresponding velocity ${V_i}^t = \left({v_{x,i}}^t, {v_{y,i}}^t\right)$.

FITNESS EVALUATION USING CROSS ENTROPY LOSS FOR EACH PIXEL ACTIVATION

The fitness of each particle is computed using the categorical cross entropy loss based on the pixel's contribution to the network's classification output. This measures how strongly the pixel activation supports accurate prediction:

$$f\left({X_i}^t\right) = \mathcal{L}_{CE}(\hat{y}_i, y) \qquad (3)$$

Where,
${X_i}^t = ({x_i}^t, {y_i}^t)$ is the position of particle i at iteration t,
$\hat{y}_i$ is the model's predicted output using the activation at $({x_i}^t, {y_i}^t)$

y is the true class label for the input image,
$\mathcal{L}_{CE}$ is the categorical cross entropy loss.
Particles with lower loss values are considered more informative for the classification task and thus move toward more optimal spatial regions.

PSO UPDATE IN BOTH DIRECTIONS

The particle positions are then updated using the standard PSO velocity and position equations, applied in both the x and y directions. This process is repeated over several iterations, during which each pixel shifts toward its optimal spatial location within the feature map.

FINAL OPTIMIZED FEATURE MAP

After convergence, the refined positions of all particles form an updated feature map. This optimized map is then passed to the fully connected layers for classification. It captures spatially meaningful activations that have been optimized to reduce classification loss, effectively improving the overall learning performance of the network.

At the first iteration, the personal best (pbest) of each particle is initialized to its current location. Subsequently, the velocity and position of each particle are updated iteratively. This iterative process continues until the desired number of iterations is reached. The feature maps extracted through this PSO-operated layer serve as inputs for the subsequent classification process. In summary, the PSO-based feature extraction layer enhances the internal representation of the feature maps, optimizing pixel values through iterative updates and fitness evaluations, and thereby contributing to improved classification performance.

TABLE 1. Detailed description of each component of the proposed NN architecture

Layer	Input Dimensions	Operations/Details	Output Dimensions
Input Layer (MNIST data)	28×28× 1	Input image	28× 28 × 1
Convolution layer	28×28× 1	16 filters, 3 × 3 size, stride=1, padding =0	28× 28 × 1
Group convolution	28× 28 × 1	Groups =2, 3 × 3 size	24 × 24 × 8
PSO based layer	24 × 24 × 8	PSO optimized layer	12 × 12 × 8
Flatten layer	12 × 12 × 8	Flatten the layer	12 × 12 × 8
Fully connected layer 1	12 × 12 × 8	256neurons, activation-ReLU	256
Fully connected layer 2	256	64 neurons, activation-ReLU	64
Output layer	64	10 neurons, activation - Softmax	10

ANALOGY BETWEEN PSO AND FEATURE MAP OPTIMIZATION

SPATIAL COORDINATES AS PARTICLES

In this analogy, we treat each spatial coordinate within the feature map as a particle in the PSO algorithm. These spatial coordinates represent potential solutions or positions in the search space that we want to optimize.

PIXEL VALUES AS FITNESS SCORE

The pixel value at a particular spatial coordinate within the feature map represents some information or characteristic of the input image or data that the CNN is processing. It could be the intensity value of an image pixel or the activation value of a neuron in the feature map. This value can be considered analogous to the fitness or evaluation score of a particle in PSO.

OPTIMIZATION OBJECTIVE

The aim of the optimization process is to identify spatial coordinates that yield high-quality pixel values based on specific criteria. For example, we might want to find spatial coordinates that correspond to regions of high activation in the feature map, indicating the presence of important features in the input data.

PSO UPDATES

During the PSO optimization process, particles (spatial coordinates) adjust their positions iteratively based on their velocities and the influence of their personal best and global best positions. The fitness or evaluation score associated with each particle (pixel value) guides this adjustment, just as it guides the movement of particles toward better solutions in traditional PSO.

FITNESS FUNCTION

The fitness function in this context evaluates the quality of each spatial coordinate based on its corresponding pixel value. This function determines how well a particular spatial coordinate contributes to achieving the optimization objective. The fitness function evaluates the performance of the neural network given the current set of pixel values in the feature map. This typically involves computing the loss function.

MATHEMATICAL DERIVATIONS

Velocity update equation:

$$V_{x,i}{}^{t+1} = \omega . V_{x,i}{}^{t} + c_1 r_{1.}\left(P_{x,i}{}^{best} - x_i{}^{t}\right) + c_{2.} r_{2.}\left(g_x{}^{best} - x_i{}^{t}\right) \quad (4)$$

$$V_{y,i}{}^{t+1} = \omega . V_{y,i}{}^{t} + c_1 r_{1.}\left(P_{y,i}{}^{best} - x_i{}^{t}\right) + c_{2.} r_{2.}\left(g_y{}^{best} - x_i{}^{t}\right) \quad (5)$$

Position update equation:

$$X_i{}^{t+1} = X_i{}^{t} + V_{x,i}{}^{t+1} \quad (6)$$

$$Y_i{}^{t+1} = Y_i{}^{t} + V_{y,i}{}^{t+1} \quad (7)$$

Where,

$V_{x,i}{}^{t+1}$ = particle's velocity at x direction,
$V_{y,i}{}^{t+1}$ = particle's velocity at y direction,
$X_i{}^{t+1}$ = particle's position at x direction,
$Y_i{}^{t+1}$ = particle's position at y direction.

In the PSO-operated layer, particle movement follows the velocity and position update rules in Eqs. (4) and (5). The velocity update combines three factors:

- an inertia term $\omega . V_{x,i}{}^{t}$, which preserves part of the particle's previous velocity.

- a cognitive term $c_{1.}r_{1.}\left({P_{x,i}}^{best} - {x_i}^t\right)$, which directs the particle toward its own best position found so far, and
- social term $c_{2.}r_{2.}({g_x}^{best} - {x_i}^t)$, which attracts the particle toward the global best position found by the swarm.

The position update then adjust the particle's location based on its new velocity, ensuring that it remains within the bounds of the feature map. This mechanism enables each particle to balance exploration of new regions and exploitation of the best solutions identified, improving feature optimization within the CNN.

ALGORITHM: PSO BASED FEATURE EXTRACTION LAYER

1. Initialization
   - Number of swarms = 8
   - Particle Initializations:
     - Initialize N particles with random positions $(x_{i,}y_i)$ within the bounds of the feature map.
     - Initialize velocities V_i for each particle.
   - Initialize best values:
     - Set personal best positions $pbest_i = (x_i, y_i)$ to the initial positions.
     - Set personal best fitness as $pbest_{fitness}$
     - Set global best position $gbest$ to the position with the best fitness value among all particles.
     - Set global best fitness as $gbest_{fitness}$

2. Iterative Optimization Process
   - For each iteration
     - Fitness Evaluation:
       - For each particle *i*
         - Evaluate the fitness at its current position (x_i, y_i):
           $$fitness_i = Loss\left(activation\ value\ (x_i, y_i)\right):$$
       - End for
     - Update Personal Best(pbest):
       - For each particle *i*
         If $fitness_i < pbest_{fitness}$
         - Set $pbest_i = (x_i, y_i)$
         - Set $pbest_{fitness} = fitness_i$

         End if

       End for
     - Update Global Best (gbest):
       - Identify the particle with the best fitness in the current iteration:
         - If $\min(fitness_i) < gbest_{fitness}$

- Set gbest = (x_i, y_i) where $fitness_i = min(fitness_i)$
- Set $gbest_{fitness} = min(fitness_i)$
- End if

- Update Velocity and Position:
  - For each particle *i*
    - Update the velocity:

$$V_{x,i}{}^{t+1} = \omega . V_{x,i}{}^{t} + C_1 . r_1 . \left(pbest_{x,i} - x_i{}^{t}\right) + C_2 . r_2 . (gbest_x - x_i{}^{t})$$

$$V_{y,i}{}^{t+1} = \omega . V_{y,i}{}^{t} + C_1 . r_1 . \left(pbest_{y,i} - y_i{}^{t}\right) + C_2 . r_2 . \left(gbest_y - y_i{}^{t}\right)$$

    - Update the position:

$$X_i{}^{t+1} = X_i{}^{t} + V_{x,i}{}^{t+1}$$

$$Y_i{}^{t+1} = Y^{t} + V_{y,i}{}^{t+1}$$

    - End for

- Ensure particles stay within the bounds

- Repeat until convergence criteria are met.

- End for

3. Output
   - The global best position and fitness provide the optimal solution with the minimum loss.
4. End

IMPLEMENTATION DETAILS

The essential implementation aspects for this proposed approach are shown in Table 2.

TABLE 2. The important key features

Category	Details
Programming Language	Pytorch
GPU	*Model: Tesla T4, Number of GPU:1, Memory 14.75GB, Cuda capability:7.5*
CPU	*Processor:×86_64, Memory:12.67GB*
RAM	*1.03 GB*

TRAINING PROCEDURE

An essential aspect of the suggested technique in the PSO algorithm is initialized of values. Table 3 displays the initial values for the algorithm. Figure 2 illustrates the comprehensive flow chart of the entire procedure.

TABLE 3. Parameters and their initialized values used in the algorithm

Parameter	Values
Number of swarms	8 swarms
No.of particles	No.of features in the datasets
Inertia coefficient - ω	$0.9 - 0.01 \times iteration$
C_1	2.0(Empirical Analysis)
C_2	2.0(Empirical Analysis)
Maximum number of iterations	10
Position	The geometric position of the feature of the feature map
Velocity	Randomly generated
pbest	Local best position of particles – Initially current position
gbest	Global best position of the overall particles.

To thoroughly evaluate the effectiveness of the proposed EPSO-based optimization, we conducted comparative experiments using two architectural variants. The Proposed model, which removes traditional pooling layers and incorporates a PSO operated layer, was trained using the EPSO strategy. For comparison, a baseline model was constructed by replacing the PSO-operated layer with conventional pooling layers and training it using three standard gradient based optimizers: SGD, Adam and AdaDelta. All models were trained under identical conditions including batch size, learning rate, number of epochs, and dataset splits on three widely adopted benchmark datasets: MNIST, CIFAR10 and CIFAR100. This uniform experimental setup ensures a fair comparison across both architectural and optimization strategies.

TABLE 4. Summary of Dataset used in the Experiments

Dataset	No.of classes	Image dimensions	Training images	Testing images
MNIST	10	28 × 28	60,000	10,000
CIFAR-10	10	32 × 32	50,000	10,000
CIFAR-100	100 (20superclasses)	32 × 32	50,000	10,000

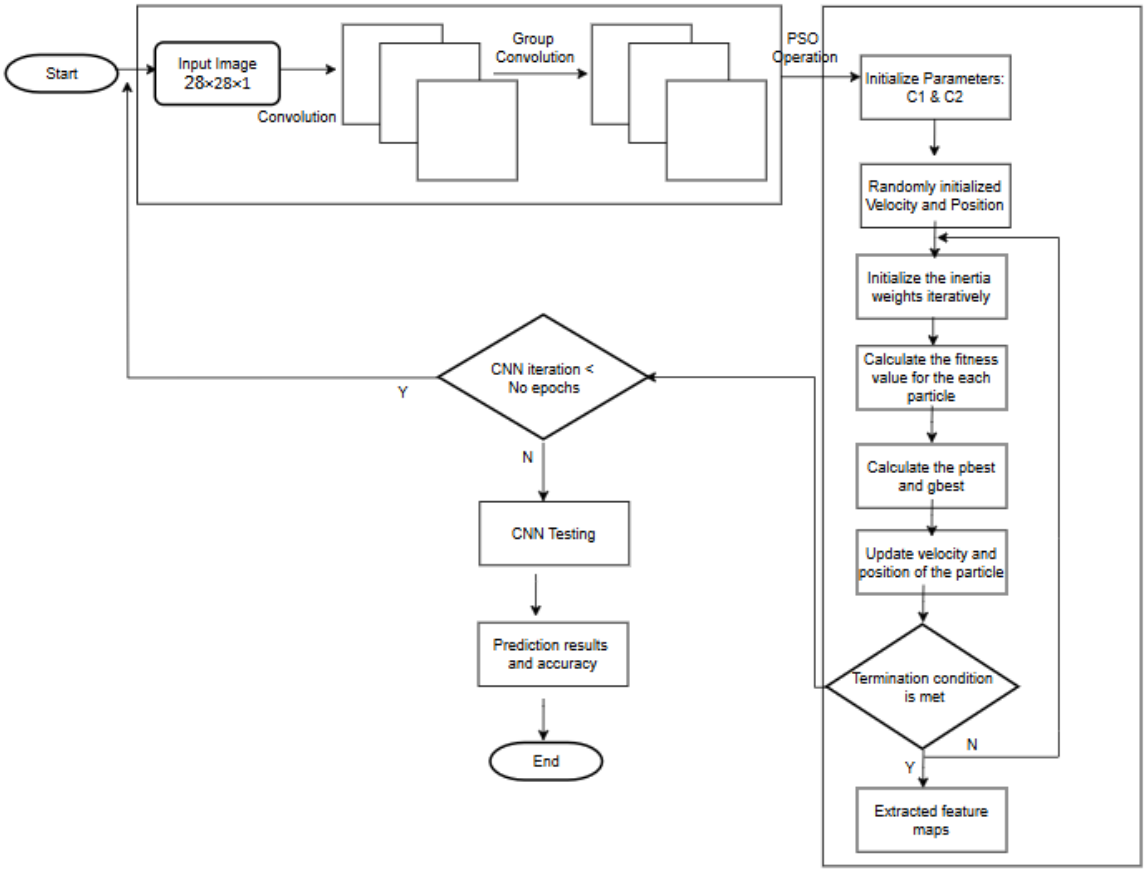

FIGURE 2. The flowchart of the proposed approach

RESULTS AND DISCUSSION

To evaluate the classification accuracy of the proposed approach, two distinct experiments were conducted. In the first experiment, a PSO-based layer was integrated into a DCNN. This novel architecture, termed enhanced PSO for CNN (EPSO_CNN), aimed to enhance the feature extraction process by optimizing the activation values through PSO operations. In the second experiment, a baseline model based on the standard LENET-5 architecture was employed, which utilized conventional pooling layers, convolutional layers, and flatten layers. This baseline was trained under three widely used optimization strategies: SGD, Adam, and AdaDelta. This setup allowed for a fair comparison between the proposed EPSO_CNN and traditional gradient-based optimization approaches.

The performance comparison between the proposed model and the LENET-5 model on the MNIST dataset is shown in Table 5. Table 6 represents a detailed comparison of the performance of both models on the CIFAR- 10 dataset. Furthermore, Table 7 indicates the performance results of the proposed model and LENET-5 model on the CIFAR-100 dataset, incorporating adjustments made during the experimental trials.

TABLE 5. Classification accuracy (%) over 10 epochs on MNIST for EPSO-CNN and baseline CNN models trained with SGD,Adam, and AdaDelta

Epoch	EPSO-CNN Proposed Model	SGD-Baseline ModelBas Model	Adam-Baseline Model	AdaDelta-Baseline Model
1	98.52	97.87	97.92	97.68
2	98.37	98.30	98.40	98.19
3	98.62	98.52	98.55	98.41
4	99.21	98.73	98.65	98.48
5	99.21	98.34	98.70	98.51
6	98.96	98.38	98.71	98.52
7	99.1	98.82	98.71	98.53
8	99.15	98.86	98.72	98.55
9	99.14	98.77	98.72	98.57
10	99.10	98.84	98.72	98.58

TABLE 6. Classification accuracy (%) over 10 epochs on CIFAR-10 for EPSO-CNN and baseline CNN models trained with SGD,Adam, and AdaDelta

Epoch	EPSO-CNN Proposed Model	SGD-Baseline ModelBa	Adam-Baseline Model	AdaDelta-Baseline Model
1	39.17	19.98	23.20	21.70
2	44.15	33.32	36.48	34.21
3	48.12	42.09	44.01	41.65
4	52.45	45.94	49.27	47.18
5	61.23	48.51	53.66	50.91
6	65.82	50.77	56.78	53.60
7	72.14	53.57	59.34	55.49
8	78.95	55.50	60.84	57.13
9	84.56	56.33	62.21	58.49
10	88.16	57.92	62.84	60.77

TABLE 7. Classification accuracy (%) over 10 epochs on CIFAR-100 for EPSO-CNN and baeline CNN models trained with SGD,Adam, and AdaDelta.

Epoch	EPSO-CNN Proposed Model	SGD-Baseline Model	Adam-Baseline Model	AdaDelta-Baseline Model
1	13.08	26.56	10.51	9.62
2	17.45	38.85	14.63	13.61
3	22.65	42.92	19.82	18.01
4	24.87	49.03	23.65	21.34
5	32.14	51.31	28.74	25.10
6	38.52	54.05	32.92	28.56
7	40.12	55.15	35.84	30.34
8	48.13	57.23	38.89	32.50
9	58.78	57.97	40.45	34.27
10	62.65	58.58	41.96	36.03

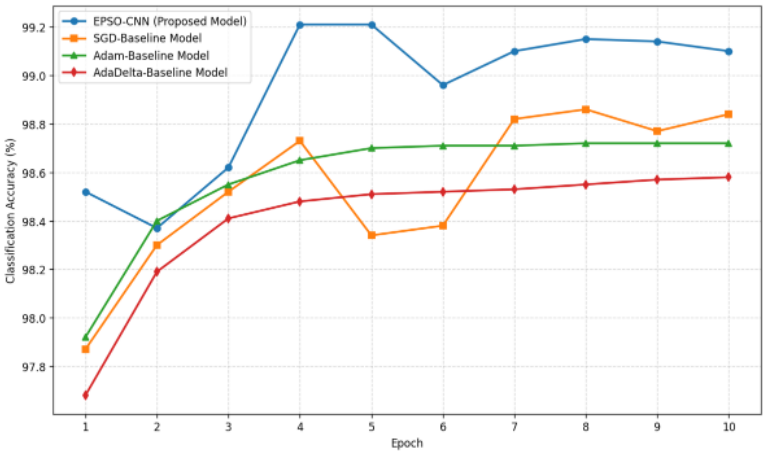

FIGURE 3. Classification accuracy (%) across 10 training epochs on MNIST

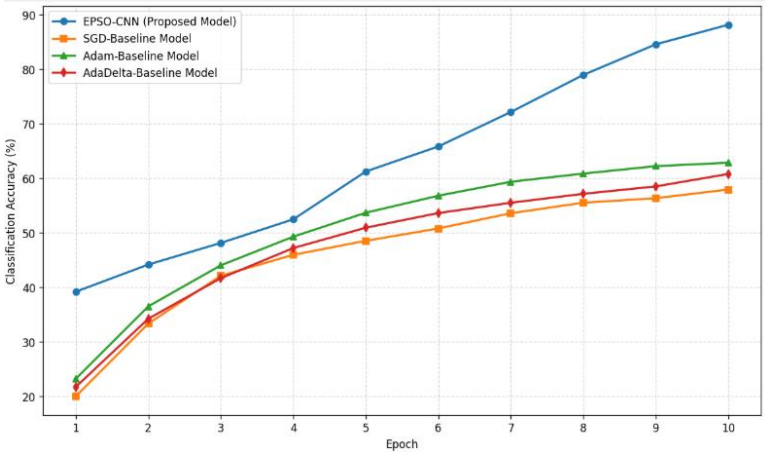

FIGURE 4. Classification accuracy (%) across 10 training epochs on CIFAR-10

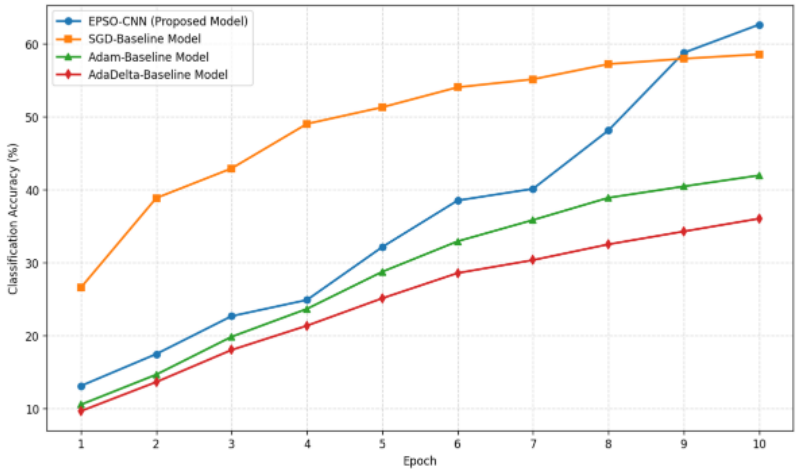

FIGURE 5. Classification accuracy (%) across 10 training epochs on CIFAR-100.

Figure 3 illustrates the classification accuracy performance of the LENET-5 model when trained with EPSO and SGD. At the 10th epoch, the accuracy reaches 99.1% with EPSO training. Figure 4 demonstrates the performance of the LENET-5 model on the CIFAR-10 dataset. When compared to the SGD training method, the accuracy with EPSO training increases to 88.16% at the 10th epoch. Figure 5 depicts the model's performance on a larger dataset, such as CIFAR-100. At the 10th epoch, the performance with EPSO training is significantly higher compared to the conventional SGD training procedure.

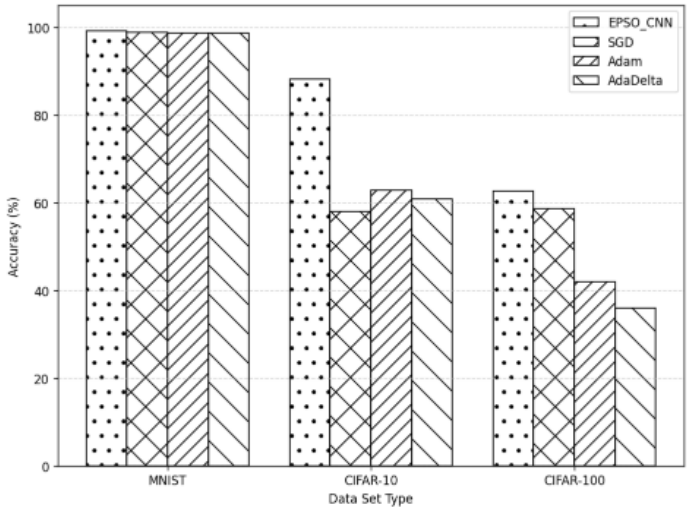

FIGURE 6. Comparison of classification accuracy (%) of EPSO-CNN and baseline CNN across MNIST, CIFAR-10, and CIFAR-100.

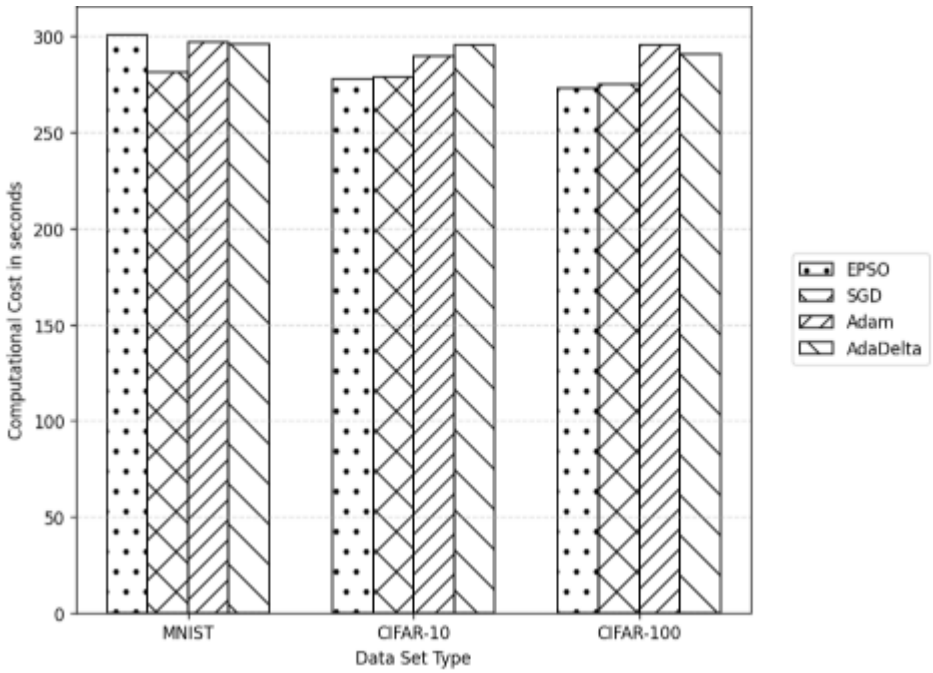

FIGURE 7. GPU utilization during model training on MNIST,CIFAR-10,and CIFAR-100 with batch size 32.

Figure 7 presents the computational cost for three types of datasets (MNIST, CIFAR-10, and CIFAR-100) when the batch size is set to 32. The computational cost is measured in terms of the time and resources required for training the LENET-5 model using the EPSO method compared to the traditional methods.

According to the results, the EPSO approach exhibits comparatively lower computational cost than the baseline methods for most datasets, though the degree of improvement varies with the optimizer employed. This observation can be explained scientifically as follows:

1. **Optimization Efficiency**: The PSO-based layer in the EPSO method optimizes the network weights more efficiently by utilizing the collective behavior of particles, which search for the optimal solution in a more directed manner. This reduces the number of iterations required to converge to an optimal or near-optimal solution, thereby lowering the overall computational cost.
2. **Convergence Rate**: EPSO tends to have a faster convergence rate due to its global search capability, which avoids local minima more effectively than SGD. This results in fewer epochs needed to achieve high accuracy, reducing the total computation time.
3. **Resource Utilization**: The EPSO method effectively leverages parallel processing capabilities of GPUs, which enhances its computational efficiency. Each particle in the swarm can be processed in parallel, making better use of the GPU resources compared to the sequential updates in the SGD method.
4. **Adaptive Learning**: EPSO dynamically adjusts the learning process based on the performance of the particles, leading to a more adaptive and potentially quicker learning phase. This adaptiveness contributes to reduced computational requirements as the learning process becomes more efficient.

The results indicate that the EPSO approach improves classification accuracy while reducing computational cost, suggesting that it is a more efficient and scalable strategy for training deep neural networks across multiple datasets.

ABLATION STUDY: IMAPCT OF PSO LAYER AND GROUP CONVOLUTION

To assess the individual contribution of the key architectural components in the proposed EPSO-CNN namely the PSO operated layer and the group convolution we conducted a series of ablation experiments. In each experiment one component was removed while keeping all other aspects of the network unchanged, including data splits, training parameters, optimizer settings and initialization. By isolating these components in a controlled manner, we can determine how much each contributes to the overall classification accuracy on MNIST, CIFAR-10 and CIFAR-100 datasets.

TABLE 8. Ablation study of the PSO-operated layer and group convolution on classification accuracy (%) for MNIST, CIFAR-10, CIFAR-100. Accuracy drop is reported in percentage points relative to EPSO-CNN

Variant	MNIST (%)	CIFAR-10(%)	CIFAR-100(%)	Accuracy Drop(PP) vs. EPSO-CNN
Baseline CNN(Without PSO)	98.34	57.92	58.58	0.76/30.24/4.07
No Group Convolution	98.52	78.41	60.21	0.58/9.75/2.44
EPSO-CNN	99.10	88.16	62.65	-

Note: Accuracy drop values are reported in percentage points relative to EPSO-CNN.

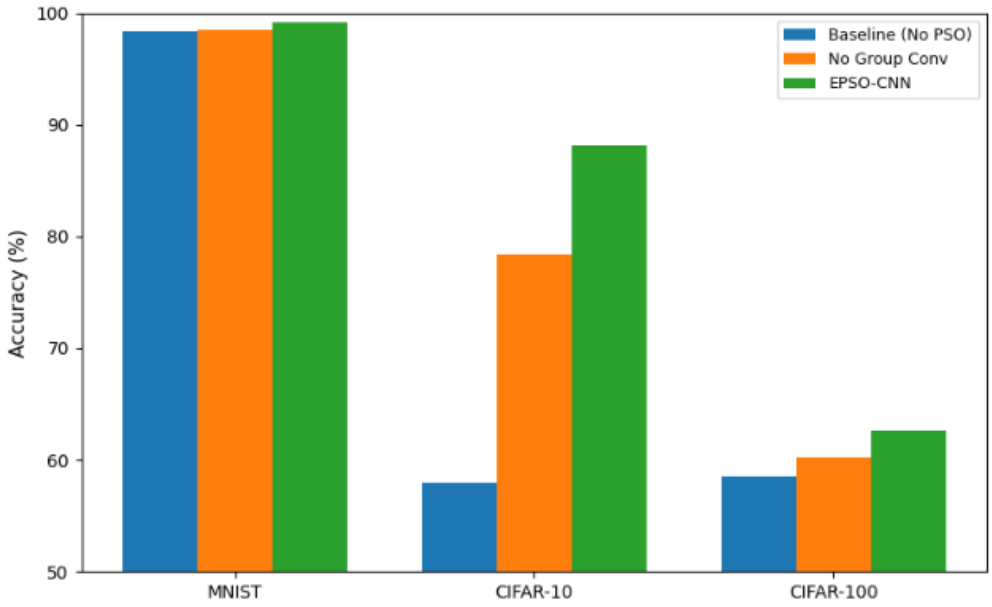

FIGURE8. Ablation study results for the PSO-operated layer and group convolution across MNIST, CIFAR-10, and CIFAR-100.

The results show that the PSO operated layer provides the largest accuracy gain, particularly on the more complex CIFAR-10 dataset, where its removal led to a drop of 30.24. Group convolution also contributes meaningfully, with a 9.75 drop in CIFAR-10 and smaller consistent improvements on MNIST and CIFAR-100. The combination of both components in the EPSO-CNN consistently achieves the highest performance across all datasets, confirming that both the PSO –operated layer and group convolution are integral to the architecture's effectiveness.

TRAINING DYNAMICS ANALYSIS

To further evaluate the effectiveness of the proposed EPSO-CNN beyond final accuracy scores, we analyzed the training dynamics of both EPSO-CNN and the baseline CNN trained with SGD. Learning curves of validation accuracy across 10 epochs are presented for MNIST, CIFAR-10 and CIFAR-100 to illustrate convergence behavior, optimization stability and overall generalization trends.

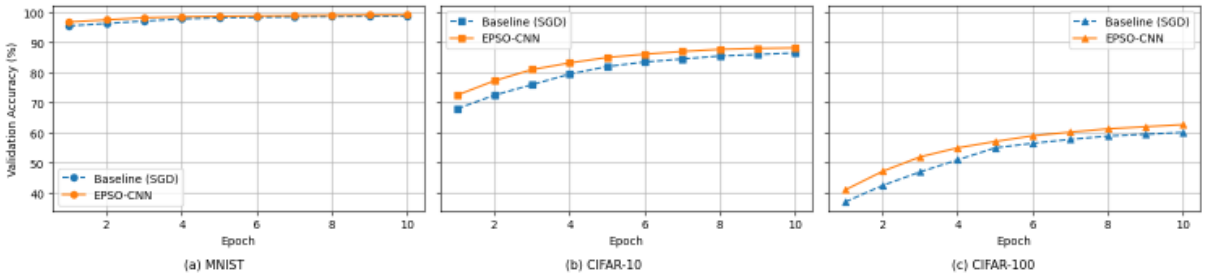

FIGURE 9. Validation accuracy versus epoch for EPSO-CNN and baseline CNN on (a) MNIST (b) CIFAR-10 (c) CIFAR-100.

The learning curves in Figure 9 provide deeper insight into the training dynamics of the proposed EPSO-CNN in comparison with the baseline CNN trained using SGD.

On MNIST, both models reach high validation accuracy, but EPSO-CNN demonstrates a slightly faster convergence, achieving above 98% accuracy by the third epoch, while the baseline requires approximately five epochs. The difference in final accuracy is modest (99.1% vs.98.7%), yet the smoother and earlier stabilization of EPSO-CNN indicates improved optimization stability.

For CIFAR-10, the performance gap is more pronounced. EPSO-CNN consistently outperforms the baseline at every epoch, with an especially strong lead during the early training phase. By epoch 5, EPSO-CNN surpasses 85% validation accuracy, compared to the baseline's 82%. At epoch 10, EPSO-CNN achieves 88.2% vs.86.5% for the baseline, underscoring its capacity to enhance convergence speed and generalization in moderately complex datasets. The most substantial improvement is observed on CIFAR-100, a considerably more challenging dataset with 100 fine-grained categories. EPSO-CNN achieves a 2.6% higher final accuracy (62.7% vs.60.1%) and consistently maintains an advantage across epochs. The faster ascent of EPSO-CNN in the early epochs suggests that the swarm based feature optimization layer enables the model to extract discriminative patterns more effectively in high dimensional classification tasks, where SGD often struggles with slower convergence and suboptimal local minima.

Collectively, these curves demonstrate that the proposed EPSO-CNN not only improves final classification accuracy but also enhances training efficiency and stability. The benefits are most evident on more complex datasets, validating the role of the PSO operated layer in guiding feature optimization beyond what is achievable with conventional gradient based methods.

CONCLUSION

In this study, we investigated the integration of an EPSO mechanism within a CNN architecture by embedding a PSO-operated layer after the group convolution stage. This design allowed direct optimization of intermediate feature maps, improving both convergence behavior and classification performance. Experimental results on MNIST, CIFAR-10, and CIFAR-100 demonstrated that EPSO-CNN consistently achieved faster convergence, greater training stability, and higher final validation accuracy compared with a baseline CNN trained using SGD.

Through ablation studies, we further isolated the contribution of the PSO layer and group convolution, confirming that both components jointly enhanced the network's representational power. The learning curve analysis also highlighted EPSO-CNN's advantage in reaching higher accuracy within fewer epochs, with improvements most evident on the more complex CIFAR-100 dataset. While the improvements represent an incremental step rather than a radical departure from existing optimization strategies, they nevertheless illustrate the potential of hybridizing evolutionary algorithms with deep learning. Further work may focus on reducing computational overhead, extending the approach to larger-scale datasets, and exploring its applicability to other domains such as segmentation and person re-identification.

ACKNOWLEDGEMENT

Thank you to Department of Computational Mathematics, University of Moratuwa for the support and facilities for this research.

REFERENCES

Ahmadzadeh, E., 2017. Optimized neural network weights and biases using particle swarm optimization algorithm for prediction applications. *Computer Science*, 20(8), pp.1406–1420.

Asmawisham Alel, M.N., Anak Upom, M.R., Abdullah, R.A. & Zainal Abidin, M.H., 2018. Optimizing blasting's air overpressure prediction model using swarm intelligence. *Journal of Physics: Conference Series*, 995(1), 012046. Available at: https://doi.org/10.1088/1742-6596/995/1/012046.

Atugoda, A.W.C.K. & Fernando, S., 2023. Improved particle swarm optimization for optimizing the deep convolutional neural network. In *Proceedings of ICITR 2023 - 8th International Conference on Information Technology Research: The Next Evolution of Digital Transformation*. Available at: https://doi.org/10.1109/ICITR61062.2023.10382832.

Bui, D.T., Nhu, V.H. & Hoang, N.D., 2018. Prediction of soil compression coefficient for urban housing project using novel integration machine learning approach of swarm intelligence and Multi-layer Perceptron Neural Network. *Advanced Engineering Informatics*, 38, pp.593–604. Available at: https://doi.org/10.1016/j.aei.2018.09.005.

Chhabra, Y., Varshney, S. & Ankita, 2018. Hybrid particle swarm training for convolution neural network (CNN). In *2017 10th International Conference on Contemporary Computing (IC3 2017)*, pp.1–3. Available at: https://doi.org/10.1109/IC3.2017.8284356.

Chhachhiya, M.G.D. & Sharma, A., 2017. Designing optimal architecture of neural network with particle swarm optimization techniques specifically for educational dataset. In *Proceedings of the 7th International Conference on Cloud Computing, Data Science & Engineering - Confluence*, pp.52–57.

de Pinho Pinheiro, C.A., Nedjah, N. & de Macedo Mourelle, L., 2020. Detection and classification of pulmonary nodules using deep learning and swarm intelligence. *Multimedia Tools and Applications*, 79(21–22), pp.15437–15465. Available at: https://doi.org/10.1007/s11042-019-7473-z.

Erivaldo, F., Junior, F. & Yen, G.G., 2019. Particle swarm optimization of deep neural networks architectures for image classification. *Swarm and Evolutionary Computation*, 49(June), pp.62–74. Available at: https://doi.org/10.1016/j.swevo.2019.05.010.

Gao, Z., Li, Y., Yang, Y., Wang, X., Dong, N. & Chiang, H.D., 2020. A GPSO-optimized convolutional neural networks for EEG-based emotion recognition. *Neurocomputing*, 380, pp.225–235. Available at: https://doi.org/10.1016/j.neucom.2019.10.096.

Ghorbani, M.A., Kazempour, R., Chau, K.W., Shamshirband, S. & Ghazvinei, P.T., 2018. Forecasting pan evaporation with an integrated artificial neural network quantum-behaved particle swarm optimization model: A case study in Talesh, northern Iran. *Engineering Applications of Computational Fluid Mechanics*, 12(1), pp.724–737. Available at: https://doi.org/10.1080/19942060.2018.1517052.

H.M., Han, T. & N.E., 2015. Hybrid algorithm for the optimization of training convolutional neural network. *International Journal of Advanced Computer Science and Applications*, 6(10), pp.79–85. Available at: https://doi.org/10.14569/ijacsa.2015.061011.

Huang, C., et al., 2019. Using a hybrid algorithm. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pp.14–19.

Khalifa, M.H., Ammar, M., Ouarda, W. & Alimi, A.M., 2018. Particle swarm optimization for deep learning of convolution neural network. In *Proceedings of the 2017 Sudan Conference on Computer Science and Information Technology (SCCSIT 2017)*, pp.1–5. Available at: https://doi.org/10.1109/SCCSIT.2017.8293059.

Kennedy, J. & Eberhart, R., 1995. Particle swarm optimization. In: *Proceedings of the IEEE International Conference on Neural Networks*. Perth, Australia, 27 November–1 December 1995. Piscataway, NJ: IEEE, pp.1942–1948.

Nandy, A., Mitra, A. & Mukherjee, T., 2020. Study of PSO and firefly algorithm based feed-forward neural network training algorithms. *In Proceedings of the 2020 7th International Conference on Signal Processing and Integrated Networks (SPIN 2020)*, pp.908–913.

Nickabadi, A., Ebadzadeh, M.M. & Safabakhsh, R., 2011. A novel particle swarm optimization algorithm with adaptive inertia weight. *Applied Soft Computing Journal*, 11(4), pp.3658–3670. Available at: https://doi.org/10.1016/j.asoc.2011.01.037.

Nithila, E.E. & Kumar, S.S., 2017. Automatic detection of solitary pulmonary nodules using swarm intelligence optimized neural networks on CT images. *Engineering Science and Technology, an International Journal*, 20(3), pp.1192–1202. Available at: https://doi.org/10.1016/j.jestch.2016.12.006.

Revathi, M., Jeya, I.J.S. & Deepa, S.N., 2020. Deep learning-based soft computing model for image classification application. *Soft Computing*, 24(24), pp.18411–18430.

Serizawa, T. & Fujita, H., 2020. Optimization of Convolutional Neural Network Using the Linearly Decreasing Weight Particle Swarm Optimization. *arXiv*. Available at: http://arxiv.org/abs/2001.05670.

Tao, Y., Shi, M.L. & Lam, C., 2019. Classification of angiosperms by gray-level co-occurrence matrix and combination of feedforward neural network with particle swarm optimization. In *International Conference on Digital Signal Processing (DSP)*, pp.1–5. Available at: https://doi.org/10.1109/ICDSP.2018.8631679.

Wang, B., Moayedi, H., Nguyen, H., Foong, L.K. & Rashid, A.S.A., 2020. Feasibility of a novel predictive technique based on artificial neural network optimized with particle swarm optimization estimating pullout bearing capacity of helical piles. *Engineering with Computers*, 36(4), pp.1315–1324. Available at: https://doi.org/10.1007/s00366-019-00764-7.

Zhang, Y., Wang, S. & Ji, G., 2015. A comprehensive survey on particle swarm optimization algorithm and its applications. *Mathematical Problems in Engineering*, Article 931256. Available at: https://doi.org/10.1155/2015/931256.