ICIC Express Letters
Part B: Applications
Volume **15**, Number **12**, December **2024**

ICIC International ©2024 ISSN 2185-2766
pp. **1227–1234**

MODIFIED PARTICLE SWARM OPTIMIZATION WITH RESETTING AND JUMPING TECHNIQUES FOR SOLVING HANDWRITTEN DIGITS

Chiabwoot Ratanavilisagul

Department of Computer and Information Sciences
Faculty of Applied Science
King Mongkut's University of Technology North Bangkok
518 Pracharat 1 Road, Wongsawang, Bangsue, Bangkok 10800, Thailand
chaibwoot@gmail.com; chaibwoot.r@sci.kmutnb.ac.th

Received January 2024; accepted April 2024

Abstract. *The Handwritten Digits Recognition (HDR) problem is the data classification involved in interpreting handwritten digit inputs from many sources, such as paper, images, touch screens, and diverse devices. This problem is the uniqueness of everyone's handwriting with many crucial characteristics. It is difficult to classify data. Many researchers try to solve this problem and propose many feature extraction techniques applied with particle swarm optimization to solving the handwritten digits recognition problem. The feature extraction process affects the search of PSO. Hence, this research proposed the feature extraction by using histogram of oriented gradients technique. Moreover, this research proposed resetting and jumping techniques integrated into the process of PSO to increase searching performance of PSO and create superior models. It can yield higher recognition rates in both training and testing phases. The proposed technique was tested on the standard MNIST datasets and individual handwritten digit datasets. It provided more satisfactory results in comparison with other PSO techniques.*
Keywords: Handwritten digits recognition, Particle swarm optimization, Trapping in local optimum, K-means, Histogram of oriented gradients, Feature extraction, Machine learning

1. **Introduction.** Handwritten digits recognition poses a significant challenge in the field of pattern recognition due to the inherent variability in people's handwriting [1]. Each individual's handwriting carries unique traits influenced by personal style and conditions at the time of writing. Consequently, even if the same individual writes the same number twice, the results might not match [2]. These intricacies make the task of HDR very difficult. The Modified National Institute of Standards and Technology (MNIST) database [3] is a well-known handwritten datasets and it is used to compare classification performance of algorithms.

In the present, HDR is mostly solved by using Deep Learning (DL) algorithms [4-6]. Their experiment results show that DL achieved a high level of accuracy in modeling. However, implementing DL in practical application is unsuitable due to its substantial data requirements, a lot of computational resources, and time-intensive [7]. In most cases, the data used for training is limited, and the computer processing power is not very strong, so using DL in practical application is not suitable.

Recently, many researchers have recommended using heuristic algorithms and metaheuristics to solve HDR or related problems. Algorithms like Particle Swarm Optimization (PSO) [8-13] and Ant Colony Optimization (ACO) [14] have shown effectiveness in tackling highly complex problems. PSO, in particular, has shown promise and is well-suited for solving HDR [11,12]. PSO is a stochastic global optimization technique employed for

DOI: 10.24507/icicelb.15.12.1227

solving optimization problems with the primary objective of finding the global optimum of a fitness function [15]. PSO has several advantages, including rapid convergence, simplicity, and minimal parameters to adjust. When applied to the task of HDR, PSO facilitates rapid model training. PSO is a suitable choice for practical applications in HDR. Hence, this paper focuses on the utilization of PSO for HDR problem-solving. However, it comes with the drawback of a heightened risk of becoming trapped in local optima [10].

In recent times, many researchers use PSO for solving HDR, such as Ba-Karait and Shamsuddin [11]. They encoded HDR as a multi-dimensional feature space and used PSO to seek the best positions for each feature space's centroid. Their experiments using the MNIST dataset demonstrate that PSO is effective in solving HDR. Samleepant and Ratanavilisagul [12] modified the feature extraction to facilitate the use of PSO. Their proposed algorithm was tested by both MNIST datasets and datasets of individual handwritten digits contributed by 35 volunteers. In comparison to algorithms that use swift learning and minimal resource consumption like PSO, their proposed algorithm which employed PSO for training yielded superior results, including higher recognition rates. Moreover, the experimental results show that the feature extraction process is important and affects the search of PSO to create models. Consequently, numerous researchers have endeavored to evaluate the efficacy of various feature extraction techniques for solving handwriting recognition problems. For instance, Zin and Otsuzuki [16] conducted tests employing Histogram of Oriented Gradients (HOG) and Bag of Visual Words (BOVW) techniques. Their experimental results indicated that HOG achieved notably higher recognition rates than BOVW. Similarly, Guha et al. [17] examined feature extraction techniques, employing Distance-Hough Transform (DHT), HOG, and Modified Log Gabor (MLG) filter Transform. Their findings revealed that HOG attained the highest classification accuracies across multiple tested datasets. Building upon prior research, this research proposes the utilization of the HOG technique for handwriting digit recognition.

As mentioned earlier, PSO is highly prone to be stuck in local optima. After getting stuck in local optimum, searching for PSO is useless and a waste of search resources. Hence, this research proposed the particle resetting technique (resetting) to solve trapping in a local optimum. This technique is re-initial to all particles when detecting population occurred trapping, influencing the search result after trapping occurred, can be worth using search resources. This approach leads to improved search outcomes. Due to the issue of becoming stuck in local optima, the models generated during the training of PSO are seldom of quality. When applying this model during the testing phase or practical use, it yields poor recognition rates. To obtain superior models, PSO must solve and alleviate the problem of trapping in a local optimum. Furthermore, this research proposes a technique to enhance the search performance of PSO through the particle jumping technique (jumping). In this method, selected particles are transformed into the best particle, thereby improving PSO's search effectiveness by exploring superior areas. As a result, this paper proposes utilizing the HOG feature extraction technique. These methods (jumping and resetting) are integrated into the PSO process to generate improved models. The proposed technique was tested on the original MNIST datasets and individual handwritten digit datasets. It provided more satisfactory results in comparison with other PSO techniques.

The remainder of this document is structured as outlined below. Section 2 explains the background. Section 3 explains the related works. Section 4 explains the proposed method. Section 5 explains the experiment results. Section 6 concludes this research and directions for development of this research in future work.

2. Background.

2.1. Particle swarm optimization.

Each member of the particle population is known as a "particle" or "X", possessing its distinct position and velocity. These particles navigate the search space based on their velocities, the best position discovered among all particles (GBEST), and their individual best position (PBEST). PSO starts by randomly initializing particle positions and velocities, evaluating each particle's position based on the objective function of the optimization problem. In each generation, every particle updates its position and velocity using the formula below:

$$V'_{i,d} = \varpi V_{i,d} + \eta_1 rand()(P_{i,d} - X_{i,d}) + \eta_2 rand()(G_{i,d} - X_{i,d}) \tag{1}$$

$$X'_{i,d} = X_{i,d} + V'_{i,d} \tag{2}$$

where $X_{i,d}$ is the previous positions of the ith particle in the dth dimension, $X'_{i,d}$ is the current positions of the ith particle in the dth dimension, $V_{i,d}$ is the previous velocity of the ith particle in the dth dimension, $V'_{i,d}$ is the current velocity of the ith particle in the dth dimension, $P_{i,d}$ is the individual best position (PBEST) of the ith particle in the dth dimension, and $G_{i,d}$ is the global best position (GBEST) in the dth dimension. ϖ is the inertia weight, η_1 and η_2 are acceleration constants. $rand()$ denotes a uniformly generated random number within the range of $[0, 1]$.

2.2. K-means clustering.

The algorithm is a widely used unsupervised learning technique employed for clustering unlabeled data by grouping features. In the K-means method, individual data points are referred to as "objects". A collection of objects forms a "class" or "centroid vector", and both classes and objects possess specific attributes. The concept of K-means can be summarized as follows: an object is assigned to a particular class based on the smallest Euclidean distance between its attributes and those of the class. The evaluation of each class is determined using the following expression:

$$d(Z_p, m_j) = \sqrt{\sum_{k=1}^{N_d} (Z_{p,k} - m_{j,k})^2} \tag{3}$$

$$J_e = \frac{\sum_{j=1}^{N_c} \left[\sum_{\forall Z \in C_{i,j}} d(Z_p, m_j) / |C_{i,j}| \right]}{N_c} \tag{4}$$

where N_d represents the total number of attributes, N_c indicates the total number of classes, Z_p represents an object designated as p, $Z_{p,k}$ denotes attribute k within object p, m_j signifies class j, $m_{j,k}$ represents attribute k within class j, $|C_{i,j}|$ stands for the count of objects belonging to class j, and $d(Z_p, m_j)$ indicates the distance between object p and class j.

2.3. Particle swarm optimization with data clustering.

The standard PSO was initially crafted to tackle benchmark function problems. To adapt PSO for data clustering tasks, modifications are introduced in the objective function and particle positions. In this scenario, an individual particle aligns with a specific class [13]. The fitness assessment of a particle relies on Equations (3) and (4). The methodology of PSO for data clustering mirrors the standard PSO process, except that each particle encapsulates the centroids of all classes, resulting in a two-dimensional array for X_i, where $X_{i,d}$ can be depicted as $X_{i,j,k}$. In this context, 'j' denotes the class of each centroid, and 'k' represents the attributes of each class. Similarly, the velocity of each particle encompasses the centroids of all classes, presenting a two-dimensional array for V_i, where $V_{i,d}$ is reframed as $V_{i,j,k}$.

2.4. **Histogram of oriented gradients.** HOG was first proposed by [18]. The concept of HOG involves picking out the important and unique characteristics from digital number images for analysis in the next step. This research suggests using a HOG method. The details of HOG can be described as follows. The process of HOG was initially developed for pedestrian detection in static images. Its fundamental concept revolves around describing the appearance and shape of local objects in an image by analyzing the distribution of intensity gradients or edge directions. Similar to edge orientation histograms, scale-invariant feature transform descriptors, and shape contexts, HOG captures these characteristics. However, it stands out by computing this information across a dense grid of evenly spaced cells and employs overlapping local contrast normalization to enhance accuracy.

3. Related Works.

3.1. **Particle swarm optimization with handwritten digits recognition.** The algorithm introduced by [11] was the first to utilize PSO in solving HDR, known as PSOHDR. Its core concept involves several steps: converting each digital image into a thinned image, followed by feature extraction based on the direction of neighboring pixels and the count of pixels in rows and columns. Next, K-means clustering is applied to classify the data. Each class is represented as a centroid in a multi-dimensional feature space, and PSO is used to determine the optimal position for each centroid or to create a classification model. PSOHDR underwent testing on the original MNIST datasets, demonstrating promising performance and consistent behavior in digit recognition.

3.2. **Particle swarm optimization with thinning processing and checked pixel chain code processing.** The algorithm proposed by [12] is named PSOTCCC. The main concept of PSOTCCC can be summarized as follows. Each digital image undergoes preprocessing methods applied before PSO. These pre-processing steps include noise removal, image segmentation, thinning processing, and checking the pixel chain code to extract features from the images. Subsequently, PSO, the same PSO as in PSOHDR, is executed to create a classification model. PSOTCCC was evaluated using both MNIST datasets and datasets containing individual's handwritten digits, achieving a high recognition rate with reduced runtime.

4. Proposal Work.

As mentioned earlier, feature extraction is a crucial step in solving HDR. This research proposes utilizing the HOG technique for feature extraction. Termed Modified PSO with HOG (MPSOH), this method follows a process similar to PSOHDR, but incorporates HOG for feature extraction.

As mentioned earlier, when the swarm becomes trapped in a local optimum, the solution stagnates until the search is completed. This scenario is akin to reaching the end of the search process in PSO. PSOTCCC, PSOHDR, and MPSOH lack a mechanism for addressing or mitigating issues associated with being trapped in a local optimum. To address this issue, this research introduces the particle resetting technique, referred to as 'resetting'. The process involves re-initializing velocities, positions, PBEST of all particles, and GBEST when local optimum trapping occurs. The subsequent search becomes useless when PSO encounters trapping in a local optimum. Resetting facilitates continued searching after encountering a local optimum trap. So, it uses search resources more effectively and creates a model with a higher recognition rate in the training phase. An indicative sign of being trapped in a local optimum is a consistent number of consecutive GBEST values remaining unchanged.

Consequently, this paper proposes the introduction of a Resetting Period (RP) based on the consecutive GBEST values that remain unaltered. Setting a very small RP would cause particles to be reset prematurely, even before they have converged. On the other

hand, an excessively large RP would result in particles remaining stuck in local optima instead of exploring other areas. Therefore, it is crucial to find the right balance for RP, favoring a somewhat larger value because setting it too small would disrupt the PSO search process. Hence, RP is easily adjustable and straightforward to conceptualize when fine-tuning it. This research proposed adding resetting into the PSO process of MPSOH (MPSOHR).

The search process in PSO relies on randomization, which can lead to searching in unsuitable areas. This randomness can sometimes result in bad performance. To enhance the search performance of PSO, certain particles of PSO should explore the best areas based on the ongoing search results. So, when GBEST improves, some particles are replaced by GBEST in the next iteration, and these particles then explore around GBEST. A number of particles that are replaced by GBEST (jumping) are called NJ. This concept can improve the search performance of PSO and create a model with a higher recognition rate in the training phase. This technique that is mentioned is called the particle jumping technique, referred to as (jumping). This research proposed adding jumping into the PSO process of MPSOH (MPSOHJ).

Consequently, the search outcomes yield improved results. Resetting and jumping have the potential to produce a model with higher recognition rates during the training phase. Such a model, characterized by a high recognition rate during training, is likely to maintain a high recognition rate in the testing phase as well. Therefore, incorporating resetting and jumping into MPSOH leads to the creation of a superior model and achieves higher recognition rates in both the training and testing phases. This paper introduces the application of resetting and jumping in MPSOH (MPSOHRJ) to achieve better search results in the training phase and consequently improve outcomes in the test phase or practical usage. The pseudo code for MPSOHRJ is provided as follows.

```
Using feature extraction with HOG
Initial position and velocity of each particle according to Formulae (5) and (7)
Initial PBEST, GBEST
For gen = 0 to MAXGEN
  For i = 0 to N
      Evaluate particle fitness according to Formula (4)
      If F(Xi) < F(PBESTi)
          PBEST = Xi
      End If
      If F(Xi) < F(GBEST)
          GBEST = Xi
      End If
      Update particle position and velocity according to Formulae (1) and (2)
  End For
  If F(GBEST) = TGBEST
      CGBEST++
  Else
      CGBEST = 0
      TGBEST = F(GBEST)
---------------------------------------------------------------------- Particle Jumping Technique
      For p = 1 to N
          For j =1 to Nc
              For k =1 to Na
                  If NJ < rand() then
                      Set p (particle) = GBEST
                  End If
              Next k
          Next j
      Next p
----------------------------------------------------------------------
  End If
---------------------------------------------------------------------- Particle Resetting Technique
If CGBEST ≥ RP
  CGBEST = 0
  Reset GBEST, PBEST
  Initial all position of p particle according to Formula (5)
  Initial all velocity of p particle according to Formula (7)
End If
----------------------------------------------------------------------
Next gen
```

In the context of these parameters: N_a represents the number of attributes, N_c represents the number of classes, N_{ch} represents the count of input data or characters used for training, N represents the total number of particles, $F(X_i)$ corresponds to the fitness of the 'i' particle, F(PBEST$_i$) is the fitness derived from PBEST for the 'i' particle, PBEST$_i$ designates the PBEST of the 'i' particle, F(GBEST) signifies the fitness of the global best (GBEST), CGBEST denotes the consecutive number of unchanged GBEST values, TGBEST is the temporary fitness used to monitor changes in GBEST, MAXGEN is the maximum number of generations, and $a_{p,j,k}$ represents the value of attribute 'p' for class 'j' and character 'k'.

$$x_{i,j,k} = \frac{\sum_{p}^{N_{ch}} a_{p,j,k}}{N_{ch}} \tag{5}$$

$$V_{\max,d} = 0.1 \times (UL_d - LL_d) \tag{6}$$

$$V_{i,d} = -V_{\max,d} + rand() \times 2 \times V_{\max,d} \tag{7}$$

Equations (5), (6), (7) serve as initial particle equations and are used with the initial position and velocity of each particle. Equation (5) is employed to determine the initial position of each particle, and this is computed based on the average value of a characteristic of class 'j' and attribute 'k'. Equation (6) is utilized to calculate $V_{\max}$ for attribute 'd', with UL_d representing the maximum value for attribute 'd', LL_d representing the minimum value for attribute 'd', and $V_{\max,d}$ indicating the maximum velocity for attribute 'd'. Equation (7) is employed to establish the initial velocity of particles.

5. **Experiments and Results.** This research using the MNIST datasets comprises 60,000 training samples and 10,000 test samples, and it can be downloaded from [19]. The datasets of handwritten digits are contributed by forty volunteers who filled out an A4-sized paper. The scanned files are in ".BMP" format, and the paper size is 1,600 × 2,240 pixels, with each paper containing 140 characters. For training, 100 characters from each row composed of numbers 0 to 9, spanning rows 1 to 10, were utilized, while the remaining 40 characters spanning rows 11 to 14 were reserved for testing. The MNIST dataset is employed as a standard benchmark for assessing algorithm performance, whereas the dataset of handwritten digits from individuals is utilized to demonstrate the practical applicability.

The performance metrics used in the experiments are described as follows. The Average Recognition Rate (ARR) percentage is calculated as the average percentage of correctly identified predictions divided by the total number of data across all runs. ARR serves as an indicator of the algorithm's classification performance. Higher ARR values correspond to better classification performance. SD, which stands for "Standard Deviation", serves as an indicator of the reliability of a clustering algorithm. A smaller SD value signifies a higher level of reliability in the clustering results.

The parameters used in all experiments are as follows: both acceleration constants, η_1 and η_2, are set to 1.45, ϖ is set to 0.75, and the population size is 500. The maximum number of iterations is limited to 1,000. The parameters for the compared algorithms and the proposed algorithm are configured based on the recommendations outlined in the original paper [12]. The number of experiments of each dataset is set as 10 runs. For the proposed algorithm, RP is initially set to 50, and NJ is set to 0.1. These specific values for RP and NJ are determined through experimentation. The experimentation process involves starting with RP set at 1 and then incrementing it gradually up to 100, ultimately selecting the values of RP that yield the best results. Similarly, the experiments commence with NJ set at 0.01 and are incremented in increments of 0.05 up to 0.9, and the values of NJ that generate the best results are chosen. This research is conducted by a personal computer of AMD FX-8320 with 3.5 GHz CPU, 8 GB RAM and Visual C++ 2010 as the programming language.

The experimental results in Tables 1 and 2 clearly indicate that MPSOHRJ surpasses PSOHDR, PSOTCCC, MPSOH, MPSOHR and MPSOHJ in terms of classification performance, as evidenced by its superior ARR across all datasets, including both the MNIST datasets and the dataset of individual's handwritten digits. Furthermore, MPSOHRJ exhibits greater reliability, as indicated by its consistently lower SD across all datasets. Therefore, MPSOHRJ outperforms PSOHDR, PSOTCCC, MPSOH, MPSOHR and MPSOHJ in classification performance and reliability. MPSOH achieves better classification performance and reliability in both the test and training phases compared to PSOHDR and PSOTCCC. It proves that feature selection using HOG provides the best results compared to other feature selection methods in this experiment. MPSOHR and MPSOHJ achieve better classification performance and reliability in both the test and training phases compared to MPSOH. It proves that both resetting and jumping techniques can enhance the classification performance and reliability of MPSOH. Moreover, MPSOHRJ outperforms MPSOHR and MPSOHJ in classification performance and reliability of all datasets. It proves that applying resetting and jumping techniques together yields better results than using only one of them. The data presented in two tables indicate that achieving a high classification performance during the training stage has a significant influence on obtaining high classification performance in the testing stage as well. As a result, MPSOHRJ can create a good model with a high recognition rate in the training stage, which subsequently leads to excellent classification performance in the testing stage.

Table 1. Comparative SD and ARR of PSOHDR, PSOTCCC, MPSOH, MPSOHR, MPSOHJ, and MPSOHRJ on MNIST datasets

Technique		PSOHDR	PSOTCCC	MPSOH	MPSOHJ	MPSOHR	MPSOHRJ
Train	ARR	73.15%	77.57%	80.52%	89.41%	95.64%	98.15%
	SD	17.08	15.17	12.05	10.10	5.58	2.34
Test	ARR	66.58%	70.27%	74.95%	83.50%	89.72%	91.98%
	SD	21.51	19.51	16.60	15.13	10.18	8.34

Table 2. Comparative SD and ARR of PSOHDR, PSOTCCC, MPSOH, MPSOHR, MPSOHJ, and MPSOHRJ on the individual's handwritten digits datasets

Technique		PSOHDR	PSOTCCC	MPSOH	MPSOHJ	MPSOHR	MPSOHRJ
Train	ARR	95.45%	97.32%	98.80%	99.34%	99.90%	100%
	SD	3.40	0.96	0.82	0.32	0.18	0
Test	ARR	84.33%	87.60%	94.88%	95.88%	96.25%	98.38%
	SD	23.83	15.13	2.59	2.18	1.89	0.23

6. **Conclusion.** This improvement positively influences the recognition rate during the training phase and yields higher-performing models when employed for classification in the testing phase or practical usage. The experiments show that the solutions produced by MPSOHRJ outperform those obtained through comparison with other algorithms in terms of reliability and classification performance for both the standard datasets and the datasets reflecting practical usage. It proves that jumping and resetting can enhance the classification performance and reliability of PSO in the training phase, resulting in a higher recognition rate. It leads to the creation of a better model. When this model is used, it can classify data more effectively. In the future, I plan to develop MPSOHRJ to verify the results in the testing stages. The practical application does not determine

whether the prediction results are true or false. I plan to add a novel technique to verify and confirm the results during the test phases.

REFERENCES

[1] M. A. Mohamad, H. Hassan, D. Nasien and H. Haron, A review on feature extraction and feature selection for handwritten character recognition, *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol.6, no.2, 2015.

[2] M. H. Alkawaz, C. C. Seong and H. Razalli, Handwriting detection and recognition improvements based on hidden Markov model and deep learning, *2020 16th IEEE International Colloquium on Signal Processing & Its Applications (CSPA)*, pp.106-110, 2020.

[3] L. Deng, The MNIST database of handwritten digit images for machine learning research, *IEEE Signal Processing Magazine*, vol.29, no.6, pp.141-142, 2012.

[4] S. Pashine, R. Dixit and R. Kushwah, Handwritten digit recognition using machine and deep learning algorithms, *International Journal of Computer Applications*, vol.176, no.42, 2020.

[5] S. Bhattacharjee, M. B. U. Sifat, J. B. Kibria, N. S. Pathan and N. Mohammad, Recognition of Bengali handwritten digits using spiking neural network architecture, *2023 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, Chittagong, Bangladesh, pp.1-5, 2023.

[6] W. Liu, J. Wei and Q. Meng, Comparisions on KNN, SVM, BP and the CNN for handwritten digit recognition, *2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*, pp.587-590, 2020.

[7] J. Subhedar and A. Mahajan, A review on recent work on OCT image classification for disease detection, *2022 OPJU International Technology Conference on Emerging Technologies for Sustainable Development (OTCON)*, Raigarh, Chhattisgarh, India, pp.1-6, 2023.

[8] W.-C. Pu, J.-J. Yang and Y.-C. Luo, A sensorless direct torque control induction motor driver with a full-order flux observer based on particle swarm optimization parameter estimators, *ICIC Express Letters, Part B: Applications*, vol.7, no.10, pp.2275-2282, 2016.

[9] C.-T. Hsieh, H.-T. Yau, J.-S. Zheng, C.-C. Wang and H.-C. Chen, An optimized PID magnetic bearing control system based on particle swarm optimization, *ICIC Express Letters, Part B: Applications*, vol.7, no.8, pp.1693-1698, 2016.

[10] C. Ratanavilisagul and B. Kruatrachue, A modified particle swarm optimization with mutation and reposition, *International Journal of Innovative Computing, Information and Control*, vol.10, no.6, pp.2127-2142, 2014.

[11] N. O. S. Ba-Karait and S. M. Shamsuddin, Handwritten digits recognition using particle swarm optimization, *2008 2nd Asia International Conference on Modelling & Simulation (AMS)*, pp.615-619, 2008.

[12] J. Samleepant and C. Ratanavilisagul, Modified feature extraction to apply PSO for solving handwritten digits, *2023 20th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, Phitsanulok, Thailand, pp.356-361, 2023.

[13] C. Ratanavilisagul, A novel modified particle swarm optimization algorithm with mutation for data clustering problem, *2020 5th International Conference on Computational Intelligence and Applications (ICCIA)*, pp.55-59, 2020.

[14] C. Ratanavilisagul, Modified ant colony optimization for vehicle routing problem with time windows using limited search space and novel updating pheromone and re-initialization pheromone techniques, *ICIC Express Letters, Part B: Applications*, vol.13, no.6, pp.571-580, 2022.

[15] J. Kennedy and R. Eberhart, Particle swarm optimization, *Proceedings of International Conference on Neural Networks (ICNN'95)*, vol.4, pp.1942-1948, 1995.

[16] T. T. Zin and T. Otsuzuki, Usability of tablet mobile devices for offline handwritten character recognition, *ICIC Express Letters, Part B: Applications*, vol.11, no.6, pp.587-593, 2020.

[17] R. Guha, M. Ghosh, P. Singh, R. Sarkar and M. Nasipuri, A hybrid swarm and gravitation-based feature selection algorithm for handwritten *Indic* script classification problem, *Complex & Intelligent Systems*, vol.7, DOI: 10.1007/s40747-020-00237-1, 2021.

[18] N. Dalal and B. Triggs, Histograms of oriented gradients for human detection, *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, San Diego, CA, USA, vol.1, pp.886-893, DOI: 10.1109/CVPR.2005.177, 2005.

[19] *https://git-disl.github.io/GTDLBench/datasets/mnist_datasets/*, Accessed on March 6, 2024.