JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

# Improved Model Configuration Strategyusing Particle Swarm Optimization for Kannada Handwritten Numeral Recognition

**Gopal D. Upadhye[1*],**
[1]JSPM's Rajarshi Shahu College of Engineering, Pune, Maharashtra, 411033, India

## Abstract
***Handwritten numeral recognition has been an important area in the domain of pattern classification. The task becomes even more daunting when working with non-Roman numerals. While convolutional neural networks are the preferred choice for modeling the image data, the conception of techniques to obtain faster convergence and accurate results still poses an enigma to the researchers. In this paper, we present a new method for the optimization of the traditional convolutional neural network architecture to obtain better results for Kannada numeral images. Specifically, a new strategy for choosing the CNN configuration is presented, which is based on particle swarm optimization. The optimization strategy is helpful to reduce the time required for the manual iterative approach of architecture selection. The proposed setup is trained on varying handwritten Kannada numerals. The new method is assessed on multiple standard datasets: the standard dataset of Kannada Dig-MNIST and our own custom created dataset. Significant improvements across multiple performance metrics are observed in our proposed system over the traditional CNN training setup. The improvement in results makes a strong case for relying on such method for faster and more accurate training and inference of digit classification, especially when working in the absence of transfer learning.***

## Keywords
***Numeral recognition, Particle Swarm Optimization, Convolutional Neural Networks, Kannada Numerals***

## 1. Introduction

Pattern classification involves modeling of data to extract relevant features that indicate a particular pattern. Research has been actively pursued on handwritten numeral recognition, an important application of pattern classification. Given an input image, the task objective is to correctly predict one of the ten numerals to which the image belongs. Most of the contemporary methods, including state-of-the-art algorithms, work well on Latin numerals. There is still a vast scope for improvements in the performance of numeral recognition systems on non-English numerals [1], [2]. With the range of data becoming more diverse, systems working on non-English numerals are the need of the hour.

Kannada is a native Indian language with a rich history spoken in the state of Karnataka and other neighboring regions [26]. The numerals in Kannada are known for their peculiar curves and angles in the writing style [3],[4]. Further, every individual will have a different way of writing these numbers. Handwritten Kannada numeral recognition thus becomes a daunting task. The use of contemporary machine learning and neural network-based architectures has been limited in case of Kannada numeral recognition algorithms.

Majority of previous approaches in this domain have used traditional statistical feature-based methods or preliminary machine learning algorithms [5], [6]. Deep learning approaches have focused on using basic ANNs or CNNs. Advanced AI techniques are gaining adoption across multiple domains [7], [8]. Through this work, a new method has been proposed to enhance the working of convolutional neural networks for Kannada handwritten numeral recognition. Specifically, we work on the designing of an improved optimization strategy to decide the best-suited model configuration.

The three important novelties of this work can be enlisted as:

1. We have contributed a new Kannada numeral dataset to the community created under a standard setup to promote further research in this area.
2. We incorporate an evolutionary algorithm to decide the architecture setup that is otherwise derived empirically in traditional techniques. Thus, any possibility of bias is eliminated.
3. We achieve the best results on two different datasets: The standard Dig-MNIST dataset and our custom created dataset.

The paper structure is as follows: A brief overview of the previous related work and background is done in section 2. The proposed technique is described in depth in section 3. Section 4 focuses upon describing the used datasets while the obtained results are analyzed in section 5. The paper is concluded in section 6.

## 2. Literature Review

The previous approaches for Kannada languages have relied on the use of derived or handcrafted features or inputs fed to the algorithms. The contributions in the domain of deep learning, especially using advanced convolutional networks have been paltry.

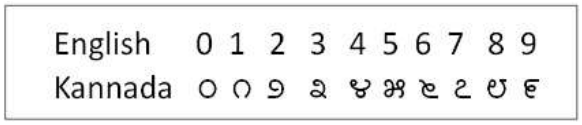


**Figure 1** Equivalence between English and Kannada numerals

One of the first approaches for this task used features based on moments to predict the numerals [9].These were passed as input to a feed-forward neural network to train the final system. Kumar presented an approach that involved splitting the input image into four sections and extracting the required components from each of these sections [10]. This approach required very less trainable parameters and inference time but this speed of prediction did take a toll on the accuracy of the system. A clustering approach using the K means algorithm was used by Sheshadri et al. [11]for conducting OCR for printed characters in Kannada. Kavya et al. [12] made use of zoning that involves feature extraction from demarcated zones, fused with a classifier to get the final output. Recognition of isolated digits was worked upon by Gurudath and Ravi [13]. Karthik et al. [4] applied SVM-based kernels and gradient descriptors for this task. Killedar and Deshpande [14] proposed a template-based matching strategy for recognition as well as translation of Kannada numbers. The nearest neighbor approach has also been tried out to produce satisfactory results. Hallur and Hegadi earlier proposed a holistic approach [15] followed by a feed-forward neural network approach for performing the task [16]. Isolated Kannada numerals were recognized using a framework built on data fusion by Mamatha et al. [17]

Dhandra et al. [18] made use of zoning to derive features for subsequent recognition of both handwritten and print numerals, and also vowels. A relatively unorthodox approach involving Fourier descriptor plates and crack codes was used by Rajputto further improve the performance [19]. Recently, deep neural networks were used by Ganesh et al. [6] for this task, a welcome improvement considering the use of recent architecture.

An observable trend in the majority of previous works is that they have focused on using primitive methods and traditional approaches, not looking beyond these concepts to use contemporary superior architectures or methods. Further, the results achieved have not been up to the mark of those achieved on English or other language numerals.

**Table 1** Comparison of previous works in terms of accuracy

| Approach | SVM | CNN | K-NN | BP-NN |
|---|---|---|---|---|
| **Dataset samples** | 1000 | 200 | 1000 | 50 |
| **Result (%)** | 97.4 | 86.28 | 91 | 95 |

## 3. Methods

The architecture configuration of CNNs is derived iteratively until the best possible configuration is not obtained. This leads to an introduction of human bias as well as an additional time requirement [20] before obtaining the best configuration of the involved layers [21]. Particle Swarm Optimization (PSO) is an evolutionary algorithm that helps to solve this issue. We propose the use of PSO as an optimization technique to derive the ideal CNN architecture.

PSO derives inspiration from the way swarms operate in nature. The algorithm itself consists of two stages that are:

1. Communication
2. Learning

Every possible architecture configuration is considered as a particle. In the communication phase, every particle transmits information to all the other particles present in the swarm. The algorithm needs to keep moving towards the best possible configuration i.e. the global minima. If at any stage, it can find a configuration better than the current one, it is rightfully following the concept of better [22],[23]. This is where the learning phase begins, and if the system can move towards a better state, eventually it will achieve the best possible state. Such a tuned system will be able to work on optimizing problems of any type.

**Algorithm 1: Algorithm for PSO optimized CNN**

**Input:** Population set PS, Personal best p, Global best g
**Output:** Classification model

1. Initialize Particle set PS
2. For each iteration
   a. For constitutent c in PS
      i. f_c = f(c)
         ii. If f_c>p: P = c;
   b. g = best c in PS
   c. for each c in PS
      i. v = v + c1 * r1 * (p - c) + c2 * r2 * (g - c)
      ii. p = p + v
3. Use the p and v values to initialize architecture
4. Use the dataset to train the derived architecture
5. Derive predictions from the trained model and return model

**Table 2** Search configuration of CNN for PSO

| Architecture element | Configuration | Search space |
|---|---|---|
| **Layer Conv2D 1** | Filters | (3,101) |
| | Size of filter | {3,5,7} |
| | Actv. Function | {Sigmoid, ReLU, tanh} |
| | Input shape | (28,28,1) |
| **Layer Max Pool 1** | Size of window | (2,2) |
| **Convolutional layer 2** | No. of kernels | (3,101) |
| | Kernel size | (7,5,3) |
| | Actv. Function | (Tanh, ReLU, Sigm.) |
| **Layer 2 – MaxPool** | Pool size | (2,2) |
| **Layer flat.** | | |
| **#1 Lay. FFNN** | Nodes | (3,101) |
| | Actv. Function | (Tanh, ReLU, Sigm.) |
| **#2 Lay. FFNN** | Nodes | (3,101) |
| | Actv. Function | (Tanh, ReLU, Sigm.) |
| **#3 Lay. FFNN** | Nodes | {[10]} |
| | Actv. Function | (Softmax.) |
| **Optimization** | Loss function | Cat. Crossentr. |

| | | |
|---|---|---|
| | Optimizer | {Adam, SGD} |
| **Parameters** | # epoc. | 10 |
| | Batch | [10,100] |
| | Verbose | {[2]} |

Various configurations of CNN would be iteratively explored to find the right one for the number recognition task. The PSO algorithm will consider each of them as a particle, and these particles are defined using two different entities or vectors. One is the position vector which focuses upon the current state coordinates of the particle. The other is the velocity vector which takes into consideration both, the intensity of the particle and also its potential direction [24],[25].

As we keep moving towards a better state, both the position vector and velocity vector of the particle are updated using the following equations:

$$X_i^{t+1} = X_i^t + V_i^{t+1} \quad (1)$$

$$V_i^{t+1} = wV_i^t + c_1 r_1 (P_i^t - X_i^t) + c_2 r_2 (G_i^t - X_i^t) \quad (2)$$

where X is the position vector for next iteration, Vindicates vector of velocity for subsequent loop, present state indicated by t, weight or rest factor of velocity shown by w, P is the personal best solution of given particle, G is the global best solution of particle in the population, c1 and c2 are the learning factors, and r1 and r2 are random weights in the range of 0 and 1.

PSO keeps adjusting all the possible solutions by modifying their vectors and keeps propagating until it does not find the best solution. Individual changes in velocity also depend upon other particles present in the swarm. The position is updated based on its velocity and current state, and also considering the distance from the personal best P and global best solutions G.

Table 2 indicates the configuration range through which the PSO optimizer searches to find the best architecture. Once these values are derived, the architecture is trained using the train set and tested on the test subset to check precision in its predictions. It can be seen that two different types of searches are possible. The first is to derive the distinct quantity as seen for filters, the second is to derive the best option from a predetermined category of options as in the case of activation functions. We have also fixed certain values as in the case of model train epochs or loss. Algorithm 1 indicates the procedure followed by the optimizer to derive the ideal configuration.

It should be noted that this configuration is not a fixed one and hence we cannot represent it diagrammatically in advance. The optimizer looks to derive these values based on empirical intermediate results. Thus, the training, in this case, differs by the fact that a heuristic-based approach has been used to achieve the ideal goal state (i.e. best possible results) whereas, in the case of traditional CNN, a pre-defined architecture has to be trained and then altered manually every time [23],[24].

The parameter values related to the updating of optimizers are as follows: the number of iterations and the number of particles is both set to 2. The inertia factor is set to the value of 0.7. While both c1 and c2 values are set to 2. A 5-fold cross-validation strategy is deployed with a split of train and test set of 4:1.

## 4. Dataset Description

The proposed approaches are evaluated on two different sets. One of them is Dig-MNIST dataset [3], which is a standard-sized publicly available Kannada numeral dataset. Secondly, we also create our dataset following standard practices to demonstrate the generalization ability of our proposed architectures.

The Dig-MNIST dataset consists of over 10000 handwritten digits in the Kannada language. It was created using handwritten digits filled by adult volunteers and later scanned to get grayscale images [3]. The images are also reduced and cropped in a way to bring down the image size to 28x28. Every Kannada numeral has 1024 images in this dataset. Some sample numerals in different font styles are depicted in Figure 2.

**Figure 2** Sample renderings of the numerals in different font styles

However, there are certain shortcomings in this dataset. The dataset size is not very vast enough thus making the training and feature extraction process difficult. While we do evaluate the approaches on the Dig-MNIST dataset, these limitations also encourage us to create a new dataset.

Kannada speakers of different age groups were asked to submit their writing of Kannada digits. A 4x3 rectangle grid on an A4-size sheet was used for the volunteers to write the ten numerals. 505 individuals volunteered for this cause, thereby creating a dataset of a total of 5050 digit instances as images. Some samples of the handwritten numerals are shown in Figure 3.

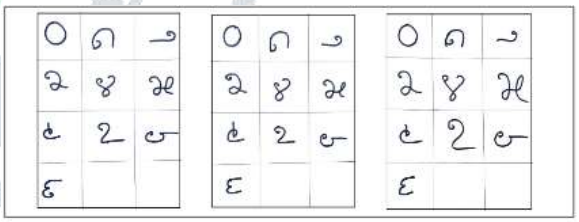

**Figure 3** Samples of handwritten Kannada digits in the grid

A couple of preprocessing steps were performed on this raw dataset. These include converting the image to grayscale format and also the application of skewing and transformations for data augmentation. Owing to these applications, we managed to quadruple the dataset size, leading to 20200 images, 2020 images for each numeral digit. We augment our created dataset further using similar steps to reach a size of 70000 images. Every image in this dataset was now of size 28x28x1, and all of them were combined and saved together in a 70000x785 sized multidimensional array in a .csv file. This dataset was more suitable for modeling.

The standard train test split was followed for both the datasets. From our created dataset, 65000 images are used for training while 5000 images are used for testing purposes. From the Dig-MNIST dataset, 8000 images were used for training while 2240 images were used for testing purposes.

# 5. Results and Discussion

## 5.1 Results on the Dig-MNIST dataset

**Table 3** Accuracy of the PSO optimized CNN on Dig-MNIST dataset

| Dig. | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | Avg |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Acc. | 87.79 | 92.79 | 95.87 | 93.40 | 88.24 | 97.24 | 83.41 | 94.3 | 92.75 | 92.48 | 91.75 |

We compare the total accuracy of the proposed approach with standard baseline methods in Table 4. Further, the class-wise accuracies for each of the ten numeral digits are tabulated in Table 3. These numbers give an overview of the generalization ability of the model.

**Table 4** Comparison of accuracy on Dig-MNIST dataset of proposed method against others

| MODEL | OVERALL ACC. (%) |
|---|---|
| Multilayer Perceptron | 86.16 |
| Decision tree | 71.97 |
| LeNet CNN | 76.58 |
| Non linear SVM | 84.23 |
| Linear SVM | 81.36 |
| CNN (random initialization) | 86.98 |
| CAE+CNN | 89.38 |
| CNN+PSO (Proposed) | 91.74 |

We have observed that the category and overall precision have improved significantly, and the overall average accuracy has exceeded 90%. It can be seen that by using PSO we have further improved the accuracy by more than 2-3%, which has helped the model to achieve a considerable improvement in performance. We also list in Table 5 the performance achieved in each fold during the PSO-optimized CNN k-fold cross-validation training.

**Table 5** Movement in k-fold accuracy during model training on the Dig-MNIST dataset

| # training fold | Accuracy |
|---|---|
| **First** | 92.76 |
| **Second** | 92.61 |
| **Third** | 98.73 |
| **Fourth** | 89.96 |
| **Fifth** | 98.54 |
| **Final** | 92.48 |

The performance of the proposed approachacross all the task-related metrics is mentioned in Table 6.

**Table 6** Results of proposed approach across all metrics on Dig-MNIST dataset

| Approach | CNN+PSO |
|---|---|
| **Accuracy (%)** | 91.75 |
| **Precision** | 0.919 |
| **Recall** | 0.917 |
| **F1 score** | 0.917 |

## 5.2 Results on the Dig-MNIST dataset

Similar to the previous dataset, the performance of the proposed approaches is calculated and analyzed on all the aforementioned parameters.

**Table 7** Class-wise accuracy of proposed approach on our created dataset

| Dig. | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | Avg |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Acc. | 94.71 | 90.93 | 90.68 | 92.84 | 91.37 | 91.08 | 92.18 | 92.7 | 89.75 | 90.46 | 91.66 |

**Table 8** Comparison of results of various approaches on our created dataset

| Classifier | Accuracy(%) |
|---|---|
| **Decision tree** | 35.10 |
| **Multilayer Perceptron** | 50.22 |
| **Linear SVM** | 33.18 |
| **Non linear SVM** | 36.64 |
| **LeNet CNN** | 73.26 |
| **CNN (random initialization)** | 88.06 |
| **CAECNN** | 89.56 |
| **CNNPSO (Proposed)** | 91.66 |

**Table 9** Variation in k-fold accuracy during training

| # Training fold | Accuracy (%) |
|---|---|
| **First** | 91.82 |
| **Second** | 92.38 |
| **Third** | 96.77 |
| **Fourth** | 89.48 |
| **Fifth** | 93.2 |
| **Final** | 93.01 |

**Table 10** Results of proposed approach across all metrics on our dataset

| Approach | CNNPSO |
|---|---|
| **Accuracy (%)** | 91.66 |
| **Precision** | 0.917 |
| **Recall** | 0.916 |
| **F1 score** | 0.915 |

Finally, for further analysis and demonstration of the performance, we have also visualized the confusion matrix for each of the proposed approaches on both the datasets.

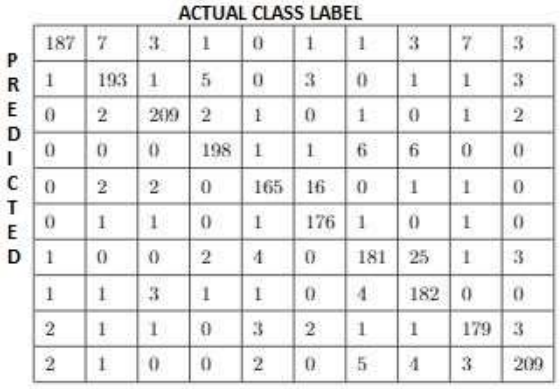


**Figure 4** Confusion matrix of proposed approach for Dig-MNIST dataset

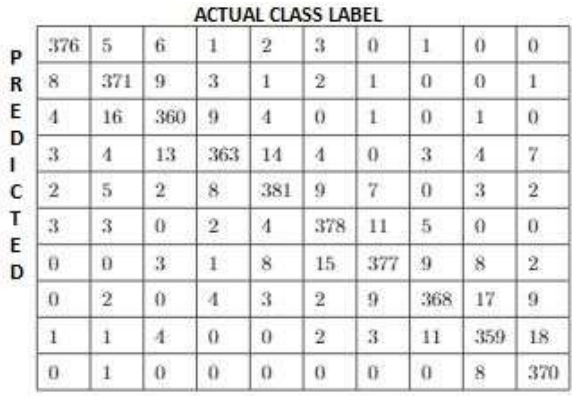


**Figure 5** Confusion matrix of proposed approach for our dataset

## 6. Conclusion and Future Scope

In this paper, a new approach was proposed for the task of handwritten Kannada numeral recognition. This approach was inspired by the need to ameliorate issues in model initialization and configuration optimization strategies. We apply the natural algorithm of particle swarm optimization that focuses on particle space search for deriving the ideal configuration of the CNN architecture. The performance of these proposed systems was compared with the standard methods traditionally used in the domain. The PSO-optimized CNN provides the best overall results with overall accuracy of over 91 while simultaneously generalizing well across all class labels. In the future, transfer learning could be applied as a technique to extend further the initialization of the CNN architecture. There has been limited work on context transfer for Kannada numerals. Ideal configurations may be provided with new member features or strategies for configuration space search to work with PSOs for much faster results. It may also improve the interpretability of these systems and provide domain users with more useful explanations for predicting beyond the values and probabilities of classifier functions.

The proposed method expects the user to provide the range in which the best configuration is to be scanned. Hence, some level of domain expertise is needed before this configuration search space can be finalized. Further, currently the architecture does not take into consideration the methods like attention mechanism, adversarial networks, or generative layers. The approach could also benefit from the inclusion of advanced learning rate variation methods including slanted triangular learning rate, gradual increase, etc.

## References

[1] A. E. Sawy, M. Loey, and H. El-Bakry, "Arabic handwritten characters recognition using convolutional neural network," WSEAS Transactions on Computer Research, Vol. 5, pp.11–19, 2017.

[2]D. T. Mane and U. V. Kulkarni, "A survey on supervised convolutional neural network and its major applications," in Deep Learning and Neural Networks: Concepts, Methodologies, Tools, and Applications, pp.1058–1071, IGIGlobal, 2020.

[3] V. U. Prabhu, "Kannada-mnist: A new handwritten digits dataset for the Kannada language," arXivpreprintarXiv: 1908.01242, 2019.

[4] S. Karthik and K. S. Murthy, "Handwritten Kannada numerals recognition using histogram of oriented gradient descriptors and support vector machines," in Emerging ICT for Bridging the Future-Proceedings of the 49th Annual Convention of the Computer Society of India CSI Volume 2, pp. 51–57, Springer, 2015.

[5] S. Shettar, B. Basavaprasad, and H. Bhagya, "Recognition of printed Kannada numerals by nearest neighbour method," in Proceedings of the International Conference on Computational Systems for Health Sustainability, pp. 99–103, 2015.

[6] A. Ganesh, A. R. Jadhav, and K. C. Pragadeesh, "Deep learning approach for recognition of handwritten Kannada numerals," in International Conference on Soft Computing and Pattern Recognition, pp.294–303, Springer,2016

[7] P. Ratadiya and D. Mishra, "An attention ensemble based approach for multilabel profanity detection," in International Conference on Data Mining Workshops (ICDMW), pp. 544–550, IEEE, 2019.

[8] P. Ratadiya, K. Asawa, and O. Nikhal, "A de-centralized aggregation mechanism for trainingdeep learning models using smart contract system for bank loan prediction," arXivpreprintarXiv: 2011.10981, 2020.

[9] L. R. Ragha and M. Sasikumar, "Adapting moments for handwritten Kannada kagunita recog-nition," 2nd International Conference on Machine Learning and Computing, pp. 125–129, IEEE, 2010.

[10] K. Prasanna Kumar, "Algorithm to identify Kannada vowels using minimum features extraction method," International Journal of Innovative Technology and Exploring Engineering, ISSN: 2278-3075, Vol. 2, Issue 2, 2013.

[11] K. Sheshadri, P. K. T. Ambekar, D. P. Prasad and R. P. Kumar, "An ocrsystem for printed Kannada using k-means clustering," IEEE International Conference on Industrial Technology, pp. 183–187, 2010.

[12] T. Kavya, V. Pratibha, B. Priyadarshini, M. Vijaya Bharathi, G. Vijayalakshmi, "Kannada characters and numerical recognition system using hybrid zone-wise feature extraction and fused classifier," Int. J. Eng. Res. Technol, Vol. 5, No. 5, 2016.

[13] K. Gurudath and D. Ravi, "Isolated digits recognition in Kannada language," International Journal of Computer Applications, Vol. 140, No. 10, 2016.

[14] S. Killedar and S. Deshapande, "Kannada handwritten numerals recognition and translation using template matching," International Journal on Recent Technologies in Mechanical and Electrical Engineering, Vol. 2, No. 6, pp. 77–80, 2015.

[15] V. C. Hallur and R. Hegadi, "Offline Kannada handwritten numeral recognition: holistic approach," Proceeding of 2nd International Conference on Emerging Research in Computing, Information, Communication and Applications, Vol. 3, pp. 632–637, 2014.

[16] V. C. Hallur and R. Hegadi, "Kannada handwritten digits recognition: neural network approach," Int. J. Sci. Res, 417-419, 2013.

[17] H. Mamatha, S. Srirangaprasad, K. Srikantamurthy, "Data fusion based framework for the recognition of isolated handwritten Kannada numerals," Int. J. Adv. Comput. Sci. Appl, Vol. 4, pp. 174–182, 2013.

[18] B. Dhandra, G. Mukarambi, M. Hangarge,"Zone based features for handwritten and printed mixed Kannada digits recognition," IJCA Proceedings on International Conference on VLSI, Communications and Instrumentation, No. 7, pp. 5–8, 2011.

[19] G. Rajput, R. Horakeri, S. Chandrakant, "Printed and handwritten Kannada numeral recognition using crack codes and Fourier descriptors plate," International Journal of Computer Application (IJCA) on Recent Trends in Image Processing and Pattern Recognition, pp. 53–58, 2010.

[20] V. Nairand, G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in ICML, 2010.

[21] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," The Journal of Machine Learning Research, Vol. 15, No. 1, pp. 1929–1958, 2014.

[22] U. V. Kulkarni, D. T. Mane, "Pattern recognition of iris flower using neural network based particle swarm optimization," International Journal of Computer Sciences and Engineering, Vol. 6, 2018.

[23] A. Roy, D. Dutta, K. Choudhury, "Training artificial neural network using particle swarm optimization algorithm," International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 3, No. 3, 2013.

[24] J. Kennedy and R. Eberhart, "Particle swarm optimization," Proceedings of ICNN'95-International Conference on Neural Networks, Vol. 4, pp. 1942–1948, IEEE, 1995.

[25] R. Eberhart, J. Kennedy, "A new optimizer using particle swarm theory," Proceedings of the 6th International Symposium on Micro Machine and Human Science, pp. 39–43, IEEE, 1995.

[26] Upadhye G. D., Kulkarni U. V., "Pattern Classification of Handwritten Kannada Digits Using Customized CNN," Advances in Intelligent Systems and Computing, Vol. 1118. Springer, Singapore. https://doi.org/10.1007/978-981-15-2475-2_56.