2023 IEEE 15th International Symposium on Autonomous Decentralized System (ISADS) | 978-1-6654-6451-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/ISADS56919.2023.10092078

Towards the Distributed Wound Treatment Optimization Method for Training CNN Models: Analysis on the MNIST Dataset

Hiram Ponce, Ernesto Moya-Albor, Jorge Brieva
Universidad Panamericana. Facultad de Ingeniería.
Augusto Rodin 498, Ciudad de México, 03920, México.
{hponce,emoya,jbrieva}@up.edu.mx

***Abstract*—Convolutional neural network (CNN) is a prominent algorithm in Deep Learning methods. CNN architectures have been used successfully to solve various problems in image processing, for example, segmentation, classification, and enhancement task. However, automatic search for suitable architectures and training parameters remain an open area of research, where metaheuristic algorithms have been used to fine-tuning the hyperparameters and learning parameters. This work presents a bio-inspired distributed strategy based on Wound Treatment Optimization (WTO) for training the learning parameters of a LenNet CNN model fast and accurate. The proposed method was evaluated over the popular benchmark dataset MNIST for handwritten digit recognition. Experimental results showed an improvement of 36.87% in training time using the distributed WTO method compared to the baseline with a single learning agent, and the accuracy increases 4.69% more using the proposed method in contrast with the baseline. As this is a preliminary study towards the distributed WTO method for training CNN models, we anticipate this approach can be used in robotics, multi-agent systems, federated learning, complex optimization problems, and many others, where an optimization task is required to be solved fast and accurate.**

***Index Terms*—Convolution Neural Network, Wound Treatment Optimization, Distributed Metaheuristic Optimization, MNIST, Image Classification, Distributed Systems**

I. Introduction

Convolutional Neural Networks (CNN) have been successfully used in object detection and classification in digital images. However, the automatic finding of adequate architectures continues still an open research area, and for many problems, it is done by hand. On the other hand, metaheuristic algorithms are techniques used to find optimal solutions to problems with limited or no knowledge, for example, the optimal hyper-parameters and learning parameters of CNN architectures and models. Thus, several metaheuristic strategies have been proposed. For example, in [1], Challapalli and Devarakonda proposed an automated fine-tuning approach through the Hybrid Particle Swarm Grey Wolf (HPSGW) method to find the optimal parameters of a CNN (batch size, hidden layers number, number of epochs, and filter size). The optimized CNN was tested on MNIST, CIFAR and Indian Classical Dance (ICD) datasets. In the MNIST case the authors obtained an accuracy of 99.4%. Raji et al. [2] presented a hyperparameters fine-tuning strategy using the Simple Deterministic Selection Genetic Algorithm (SDSGA). The proposed methodology was applied over two Machine Learning methods, a CNN architecture and the Random Forest (RF) algorithm. The optimized CNN architecture was tested over the MNIST dataset, obtaining an accuracy of 99.2%. In [3], Tuna et al. proposed the bare-bones fireworks algorithm for tuning a selected subset of hyperparameters of a CNN and tested it over the benchmark dataset MNIST. Agarwal et al. [4] developed a Genetic Algorithm (GA) based method to compress and accelerate the CNN models. Experiments using several CNN models (AlexNet, VGG16, SqueezeNet, and ResNet50) were performed using the benchmark datasets MNIST, CIFAR-10, and CIFAR-100. Wessels and van der Haar [5] reported the efficacy of Particle Swarm Optimization (PSO) to help gradient-based methods search for the optimal hyperparameters of a CNN over the MNIST dataset. In [6], Gaspar et al. compared the results of four metaheuristic algorithms: PSO, ABC (Artificial Bee Colony), ALO (Ant Lion Optimization), and BA (Bat Algorithm) to find the best solution for the hyperparameters of a CNN. Marquez Casillas and Osuna-Enciso [7] proposed a framework using the micro genetic algorithm for the automatic finding of adequate CNN architectures. The proposal was tested over three datasets: MNIST, MNIST-Fashion, and MNIST-RB, and it was compared with two frameworks from the literature, psoCNN and a simple genetic algorithm. In [8], Lee et al. reported a hyper-parameter tuning method of a CNN through the feature extraction step of CNN. The authors used a Parameter-Setting-Free Harmony Search (PSF-HS) algorithm as a metaheuristic optimization approach. In the experimentation, two simulations were performed, the first used the LeNet-5 CNN architecture and the MNIST dataset, obtaining an accuracy of 99.25%, and the second used the CifarNet CNN architecture and the Cifar-10 dataset with an accuracy of 74.76%.

In this paper, a distributed wound treatment optimization (WTO) method for training the learning parameters and accelerating the training of CNN models is proposed. WTO is a bio-inspired metaheuristic optimization method based on the behavior of the *Megaponera analis* ant from sub-Saharan Africa. To evaluate the performance of the proposed method, it was tested on the benchmark dataset for handwritten digit recognition, MNIST. We also present a distributed approach of

the WTO method to accelerate the training of a CNN model for the MNIST dataset. This distributed system consists of decentralized learning agents that can interact and communicate among them to jointly find an optimal solution to an optimization task, i.e. training the learning parameters of a CNN model. The experimental results validate our approach to be fast and accurate in the classification of handwritten digits of the MNIST dataset.

We anticipate this approach can be used in robotics, multi-agent systems, federated learning, complex optimization problems, and many others, where an optimization task is required to be solved fast and accurate.

The rest of the paper is organized as follows: in Section II, we describe the dataset used, the wound treatment optimization method, the CNN architecture, and the distributed approach using the WTO method. Section III shows the tests and results obtained. Finally, the conclusions of this work are presented.

II. Materials and Methods

This section gives a detailed description of the MNIST dataset utilized and the proposed distributed CNN-WTO based training approach.

A. Description of the Dataset

The MNIST dataset [9] was created by the National Institute of Standards and Technology (NIST), its name comes from the contraction of Modified NIST (MNIST). It consists of a collection of handwritten digits between 0 and 9. The dataset is composed for 70,000 gray-scale images of size 28×28 pixels. 60,000 images corresponds to the training set and 10,000 images as testing set.

This dataset has been used as a benchmark for training models of neural and convolutional networks. Figure 1 shows samples of each handwritten digit of the training data set, we applied a negative transformation for better visualization.

Fig. 1. MNIST image examples of the training set showing samples of each handwritten digit. A negative transformation was applied for better visualization.

B. Distributed Metaheuristic Optimization Method

In this work, we use Wound Treatment Optimization (WTO) [10], [11], an evolutionary metaheuristic optimization method. WTO is inspired by the *Megaponera analis* ant from the sub-Saharan Africa. This ant is characterized by being specialized in termite hunting. The *Megaponera analis* presents an interesting social behavior for wound treatment. During rides, the healthy ants transfer the injured ants for them to be treated.

The WTO is an evolutionary population-based metaheuristic optimization method. A set of individuals, called ants, are set first. These individuals are candidate solutions of an optimization problem that are guided by a fitness function (i.e. the objective function of the optimization problem). As proved in an earlier work [11], the WTO can be used as part of a distributed system and this is the reasoning behind choosing the WTO method for training a CNN model.

For the current work, N agents compose a decentralized and distributed system over a D dimensional search space, where each agent represents an ant. The ants can communicate with each other and their goal is to find the optimal parameters for a minimization problem with objective function f unconstrained. Let $x_i = \{x_1, \ldots, x_k, \ldots x_N\}$ be a candidate solution such that $x_i \in \mathbb{R}^D$. WTO starts initializing ants randomly and each of them is associated with a fitness function or injury function evaluation $f_i = f(x_i)$. The process is iteratively performed until it reaches a stop criterion, where the best ant g_{best} with its injury function evaluation $f(best)$ corresponds with the ant with the smallest value of f. Besides, each ant is classified according its injury level l_i: healthy ant ($S_{healthy}$), average-injured ant ($S_{average}$), and heavy-injured ant (S_{heavy}).

WTO is performed by defining the meta-parameters $\{p_{healthy}, p_{heavy}, p_{alive}\} \in [0, 1]$. $\{p_{healthy}$ and p_{heavy} percentages are used for partitioning the ant population into the sets $S_{healthy}$), $S_{average}$ and S_{heavy}, based on a sorted list of the injury function $f = \{f_1, f_2, \ldots, f_N\}$ values with $f_1 < f_2 < \cdots < f_N$, as it is shown in (1):

$$\begin{aligned} S_{healthy} &= \{f_1, f_2, \ldots, f_{k1}\} \\ S_{average} &= \{f_{k1+1}, f_{k1+2}, \ldots, f_{k2}\} \\ S_{heavy} &= \{f_{k2+1}, f_{k2+2}, \ldots, f_N\} \end{aligned} \tag{1}$$

where $k1 = \lfloor p_{healthy} * N \rfloor$ and $k2 = \lfloor p_{heavy} * N \rfloor$.

The healthy-injured ants are good candidates and they are not updated. However, heavy-injured ants are replaced by new random ants when $r > p_{alive}$, where $p_{healthy}$ is a meta-parameter involved in the exploration task and $r \in [0, 1]$ is a random number. On the other hand, average-injured ants

receive help from healthy-injured ants as it shown in (2):

$$\begin{aligned} & x_{i,n}(t+1) = \\ & h_{rate} * x_{i,n}(t) + \\ & r^1_{i,n} * \Big(x_{healthy,n}(t) - x_{i,n}(t) \Big) + \\ & r^2_{i,n} * \Big(g_{best,n}(t) - x_{i,n}(t) \Big). \end{aligned} \tag{2}$$

where $i = 1, 2, \ldots, N$, t represents the step time, $n = 1, 2, \ldots, D$ is an index for the dimensional search space, $x_{healthy}$ is a random ant take out from the $S_{healthy}$ set, $h_{rate} \in [0, 1]$ is called the help rate, and r^1, r^2 are random values in the range $[0, 1]$.

C. Convolutional Neural Networks

Within the field of image processing, there is currently an approach called Convolutional Neural Networks (CNN), which are inspired on visual perception of biological creatures [12], [13]. CNN architectures have been used successfully to solve various problems in image processing, for example, segmentation, classification, and enhancement task. Unlike other image processing techniques, CNNs do not work with previously defined features of the images. Instead, the input is the raw natural images. In other words, the CNNs are capable of working with the images without carrying out any type of prior processing. The architecture of a CNN consist of different interconnected layers. Thus, a convolutional layer applies a set of specific filters computing a feature representations of the input image, keeping only the information that is useful according to the expected results. A pooling layer, commonly connected to the output of the convolutional layer, applies a down-sampling reducing the resolution of feature maps, which decreases the number of neurons in the convolutional network and the computational complexity. Next, the output of the convolutional layer or pooling layer is passed into a rectified linear activation unit (ReLU) to increase the non-linearity of the data set, allowing to learn more complex structures in the data. As upper layers, fully-connected layers are used for classification tasks by performing high-level inferences [13]. Finally, a softmax layer is used in multi-class problems to reduce the dimensionality by assigning probabilities to each class.

In this paper, the input images to the CNN correspond to the 28×28 handwritten digits from the MNIST dataset. The CNN architecture consists of 25 filters or kernels of 12×12 for the convolutional layer. Next, the output of the convolutional layer is passed into a ReLU layer. Then, a fully connected layer with 10 outputs, one for each handwritten digit, is used for classification task. Finally, a softmax layer is used to obtain a vector of probability scores. Figure 2 shows the topology of the proposed CNN.

D. Distributed WTO for Learning CNN

For the distributed system, each agent is assumed that has computing resources so one can allocate and process sufficient amount of data. In addition, each agent can communicate to the others through a specific protocol. In this work, we assume that each agent is simulated in a separate thread in a parallel computer system. But, this system can be used in a collection of limited computing systems, e.g. embedded systems or nodes in an Internet-of-Things system. For this purpose, each agent is able to perform local computations and to broadcast its current candidate solution x_i within the fitness function value f_i. All the agents listen these two data, and locally, each one updates a table of all the performing actions of the others (a copy of this table is managed locally by each agent). Lastly, the best solution g_{best} can be extracted from this table (in any agent).

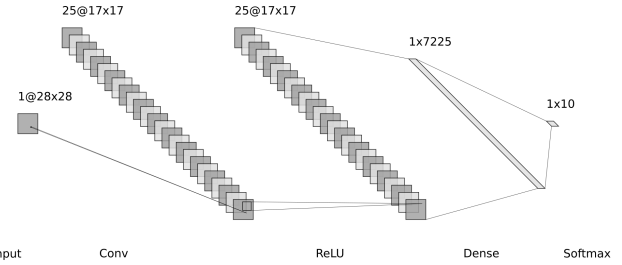

Fig. 2. Proposed CNN architecture for handwritten digit classification task.

From the above, we propose to use the distributed WTO method for learning the parameters of a CNN model. Specifically, we propose a distributed system with N agents that are related among them in a decentralized way to solve the task of finding the optimal learning parameters of the CNN proposed in Fig. 2. To do so, we set an individual x_i to be the set of learning parameters in the convolution layer, to say W_c, and in the dense layer W_d. Then, a candidate solution of agent i is $x_i = (W_c, W_d)$. Figure 3 shows the proposed distributed WTO for learning the CNN model. As shown, the optimization task is known by all the agents, and each one gets a local replica of this task. Then, each agent can find the best solution so far (i.e., the optimal learning parameters of the CNN such that it models the provided data) by computing the WTO method locally.

E. Experimental Setup

We implement our proposed distributed WTO for learning a CNN model to classify digits for the MNIST dataset in a parallel session of MATLAB using an Intel Core i7-8850H CPU at 2.60GHz with 16GB in RAM. For this purpose, each agent runs in a different thread and performing the task locally. To validate our approach, we run two experiments: (a) to compare the effectiveness between a single computer processing unit as baseline and the distributed WTO with N agents, and (b) to determine the velocity (in time) of learning using the distributed WTO depending on the number of agents involved.

For each experiment, we set up the learning process with a maximum number of 30 iterations, a maximum number of 2 epochs per iteration, and a batch size of 600. Particularly for the second experiment, we required that the CNN model reported at least 95% of accuracy to stop the learning process. A maximum of 20 iterations were set to reach the accuracy.

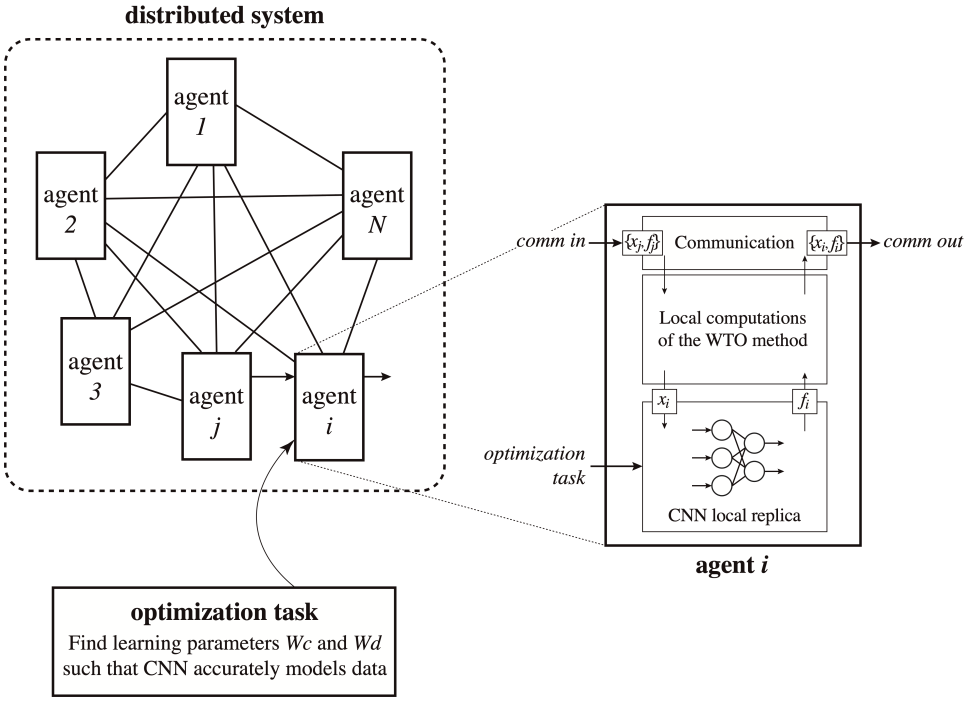

Fig. 3. Proposed distributed WTO for learning the CNN model.

For training the CNN, a 5–fold cross-validation technique was implemented using different training sizes. For convenience, we use the benchmark of training-and-testing split data reported in [9]. We choose manually the hyper-parameters of the WTO method based on previous empirical studies [10], [11], to say $p_{healthy} = 0.2, p_{heavy} = 0.3$, and $p_{alive} = 0.1$.

We evaluate the accuracy of the CNN model, in testing, using the metric shown in (3), where TP, TN, FP and FN represent the true positive, true negative, false positive and false negative values, respectively.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

Also, we measure the time the distributed system takes to train the CNN model. This training time is measured in seconds.

III. Experimental Results

This section reports the experimental results of our proposed distributed WTO for learning CNN in the MNIST dataset.

The first experiment aims to compare the effectiveness of our proposal and the baseline (using a single agent). Table I summarizes the results of this experiment in terms of the number of agents and the training percentage of data implemented, and the evaluation performance related to the test accuracy and the training time.

It can be observed in Table I that the distributed WTO system outperforms the baseline in both the test accuracy and the training time. For instance, the worst mean test accuracy of the distributed WTO (90.3080%) is 4.69% better than the best mean test accuracy of the baseline (86.2640%); while the worst mean training time of the distributed WTO (37.3867s) is 36.87% better than the best mean training time of the baseline (59.2224s). It can also be shown in Fig. 4. It is important to notice that the number of agents does not influence significantly in the mean test accuracy, but the larger the number of agents, the less variability in the performance. In contrast, the number of agents strongly impacts on the training time which drops down from 59.22s (baseline) to 9.95s ($N = 20$ agents). Furthermore, the mean test accuracy is not sensitive to the training data percentage, as shown in Fig. 4(bottom), but it is evident that a single agent is less powerful in performance than using a distributed WTO with several agents. Lastly, the mean training time is also not sensible to the training data percentage.

On the other hand, the second experiment aims to determine the velocity of training the CNN model using the distributed WTO, but also to reach at least 95% of accuracy in a maximum number of 20 iterations. Table II summarizes the experimental results. Notice that the baseline (i.e. one agent) does not reach 95% of accuracy in 20 iterations maximum, while using the distributed WTO system is fast an accurate. It is remarkable to say that these results do not mean that a CNN model cannot reach 95% of accuracy using this dataset, it is more about how many iterations do the experiment needs to reach a good accuracy.

In addition, it can be observed that the tendency to decrease the training time, as the number of agents increases, is still valid. In addition, the training time is not sensible to the training data percentage. This behavior is confirmed in Fig. 5 that shows the performance of the training time versus the

TABLE I
COMPARATIVE RESULTS BETWEEN THE BASELINE (ONE AGENT) AND THE DISTRIBUTED WTO USING DIFFERENT NUMBER OF AGENTS.

Number of agents	Training percentage	Test accuracy (%)	Training time (s)
1 (baseline)	0.10	84.9680 ± 0.8876	58.0624 ± 8.6743
	0.25	85.7140 ± 0.5865	53.6890 ± 5.4642
	0.50	86.1520 ± 0.9433	53.3044 ± 5.7889
	0.75	85.4100 ± 1.5606	53.4252 ± 4.8391
	1.00	86.2640 ± 1.7171	59.2224 ± 6.3159
5	0.10	91.2440 ± 3.6021	37.2399 ± 0.1721
	0.25	94.7180 ± 1.8575	37.3867 ± 0.1918
	0.50	94.0820 ± 2.1629	37.0055 ± 0.3182
	0.75	91.9280 ± 3.6901	36.5777 ± 0.3661
	1.00	92.9100 ± 2.6312	36.9926 ± 0.6293
10	0.10	92.4700 ± 0.7571	19.0834 ± 0.4741
	0.25	91.9720 ± 2.1344	19.1739 ± 0.3097
	0.50	92.5020 ± 2.7586	18.8945 ± 0.4479
	0.75	94.0460 ± 0.9106	18.5756 ± 0.5836
	1.00	93.2020 ± 1.0722	19.1159 ± 0.4591
20	0.10	90.5020 ± 0.4755	9.6004 ± 0.5655
	0.25	90.3080 ± 1.0720	9.0786 ± 0.0378
	0.50	90.3580 ± 1.7488	10.1440 ± 0.6157
	0.75	90.9580 ± 0.5965	9.8366 ± 0.4393
	1.00	90.3680 ± 0.4606	9.9485 ± 0.0968

TABLE II
LEARNING PERFORMANCE USING THE DISTRIBUTED WTO SYSTEM TO REACH AT LEAST 95% OF ACCURACY IN TESTING. IF THE TRAINING PROCEDURE DOES NOT REACH THE MINIMUM ACCURACY IN 20 ITERATIONS MAXIMUM, IT IS REPORTED THE BEST ACCURACY FOUND IN PARENTHESIS.

Number of agents	Training percentage	Training time (s) – (accuracy if not reached)
1 (serial)	0.10	69.4726 ± 9.5295 (84.4000 ± 2.4657)
	0.25	68.3306 ± 17.2643 (85.8020 ± 0.8773)
	0.50	62.5162 ± 13.8165 (86.3560 ± 2.1647)
	0.75	73.5231 ± 34.4959 (84.9660 ± 3.0940)
	1.00	70.3571 ± 18.5736 (86.4800 ± 1.6890)
5	0.10	48.0162 ± 10.5407
	0.25	53.3430 ± 20.2795
	0.50	47.0801 ± 10.4052
	0.75	57.6614 ± 11.2352
	1.00	42.9920 ± 7.9377
10	0.10	40.2141 ± 6.4452
	0.25	38.4781 ± 5.7633
	0.50	38.2347 ± 9.6769
	0.75	41.0328 ± 15.2360
	1.00	32.1147 ± 7.8388
20	0.10	33.5196 ± 5.9892
	0.25	30.0318 ± 3.9465
	0.50	27.7319 ± 3.0651
	0.75	26.6298 ± 1.5140
	1.00	30.2706 ± 2.7077

number of agents and the training data percentage.

From the above, the experiments valid that our proposed distributed WTO method can train a CNN model fast and accurate. Advantages found in the experiments are related to larger number of agents can accelerate the training time of the CNN model maintaining the accuracy performance, and variations in the percentage of the training data does not influence neither in the accuracy nor the training time. However, our work is limited to one dataset (i.e. the MNIST dataset) and a sensitive analysis only on the number of agents and training data percentage. Thus, more robust analysis and statistics are required to strongly validate the proposed method. But, our current work is a preliminary study towards a proposal of a distributed WTO training algorithm for CNN models.

CONCLUSIONS

This papers aimed to propose a distributed WTO method for training CNN models fast and accurate. To do so, we proposed using WTO as the core of the distributed method. We validated our approach in the MNIST dataset. The experimental results confirmed that our proposal can accelerate the training of a CNN model using more than one agent, and this improvement in the velocity can guarantee the accuracy of the model in comparison with a baseline (one agent). We anticipate this approach can be used in robotics, multi-agent systems, federated learning, complex optimization problems, and others.

For future work, we will explore the same approach but in a real multi-agent system like an IoT network to validate the proposal in a real system. Then, we will also experiment with

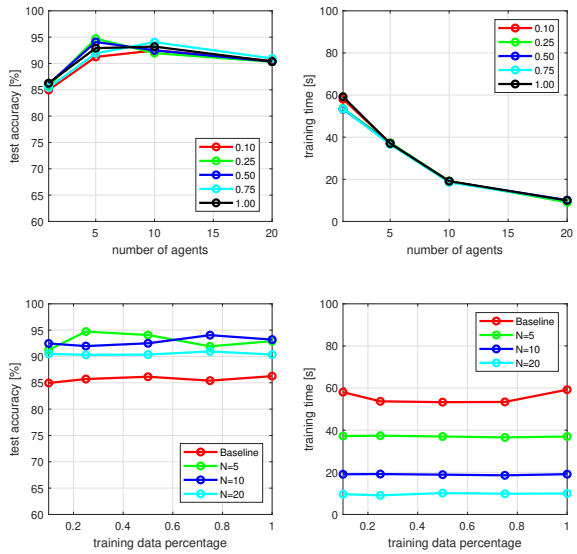

Fig. 4. Comparative results between the baseline and the distributed WTO: (top) varying the number of agents and (bottom) varying the training percentage of data.

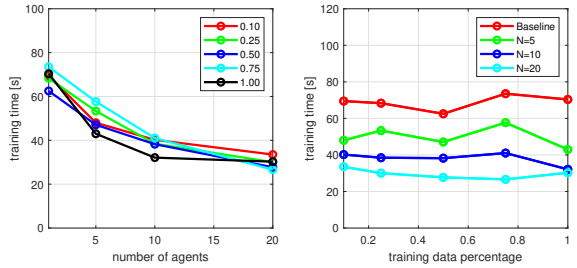

Fig. 5. Performance of the training time: (left) varying the number of agents and (right) varying the training data percentage.

other datasets and to measure the performance statistically.

ACKNOWLEDGMENT

Authors would like to thank the Facultad de Ingeniería of Universidad Panamericana for all support in this work.

REFERENCES

[1] J. Challapalli and N. Devarakonda, "A novel approach for optimization of convolution neural network with hybrid particle swarm and grey wolf algorithm for classification of indian classical dances," *Knowledge and Information Systems*, vol. 64, no. 9, pp. 2411–2434, 2022.

[2] I. Raji, H. Bello-Salau, I. Umoh, A. Onumanyi, M. Adegboye, and A. Salawudeen, "Simple deterministic selection-based genetic algorithm for hyperparameter tuning of machine learning models," *Applied Sciences (Switzerland)*, vol. 12, no. 3, 2022.

[3] E. Tuba, I. Tuba, R. Hrosik, A. Alihodzic, and M. Tuba, "Image classification by optimized convolution neural networks," *Lecture Notes in Networks and Systems*, vol. 434, pp. 447–454, 2022.

[4] M. Agarwal, S. Gupta, M. Biswas, and D. Garg, "Compression and acceleration of convolution neural network: a genetic algorithm based approach," *Journal of Ambient Intelligence and Humanized Computing*, 2022.

[5] S. Wessels and D. van der Haar, "Using particle swarm optimization with gradient descent for parameter learning in convolutional neural networks," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12702 LNCS, pp. 119–128, 2021.

[6] A. Gaspar, D. Oliva, E. Cuevas, D. Zaldívar, M. Pérez, and G. Pajares, "Hyperparameter optimization in a convolutional neural network using metaheuristic algorithms," *Studies in Computational Intelligence*, vol. 967, pp. 37–59, 2021.

[7] E. Marquez Casillas and V. Osuna-Enciso, "Architecture optimization of convolutional neural networks by micro genetic algorithms," *Studies in Computational Intelligence*, vol. 967, pp. 149–167, 2021.

[8] W.-Y. Lee, S.-M. Park, and K.-B. Sim, "Optimal hyperparameter tuning of convolutional neural networks based on the parameter-setting-free harmony search algorithm," *Optik*, vol. 172, pp. 359–367, 2018.

[9] Y. LeCun, C. Cortes, and C. J. Burges, "The mnist database of handwritten digits," http://yann.lecun.com/exdb/mnist/, 1998.

[10] H. Ponce, "Population-based metaheuristic optimization using organized social wound treatment," in *2019 IEEE 14th International Symposium on Autonomous Decentralized System (ISADS)*. IEEE, 2019, pp. 1–8.

[11] H. Ponce, E. Moya-Albor, L. Martínez-Villaseñor, and J. Brieva, "Distributed evolutionary learning control for mobile robot navigation based on virtual and physical agents," *Simulation Modelling Practice and Theory*, p. 102058, 2019.

[12] K. Nogueira, O. Penatti, and J. dos Santos, "Towards better exploiting convolutional neural networks for remote sensing scene classification," *Pattern Recognition*, vol. 61, pp. 539 – 556, 2017.

[13] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, and T. Chen, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 2017, pp. 1–24, 2017.