

# The evaluation of convolutional neural network and genetic algorithm performance based on the number of hyperparameters for English handwritten recognition

Muhammad Munsarif<sup>1,2</sup>, Edi Noersasongko<sup>3</sup>, Pulung Nurtantio Andono<sup>3</sup>, Moch Arief Soeleman<sup>3</sup>

<sup>1</sup>Graduated program of computer science, Dian Nuswantoro University Semarang, Semarang, Indonesia

<sup>2</sup>Department of computer science, University of Muhammadiyah Semarang, Semarang, Indonesia

<sup>3</sup>Department of computer science, Dian Nuswantoro University Semarang, Semarang, Indonesia

## Article Info

### Article history:

Received Oct 14, 2022

Revised Jan 1, 2023

Accepted Jan 10, 2023

### Keywords:

Convolutional neural networks

Genetic algorithms

Handwritten digit recognition

Hyperparameter optimization

## ABSTRACT

Convolutional neural network (CNN) has been widely applied to image recognition, especially handwritten English recognition. CNN's performance is good if the hyperparameter values are correct. However, the determination of precise hyperparameters is not a trivial task. This task is made more difficult when combined with a larger number of hyperparameters resulting in a high dimensionality of the search space. Usually, hyperparameter optimization uses a finite number. Previous studies have shown that a large number of hyperparameters can result in optimal CNN performance. However, the studies only apply to text mining datasets. This study offers two novelties. First, it applied 20 hyperparameters and their ranges to handwritten English. Second, this paper conducted seven experiments based on different hyperparameters and the number of hyperparameters. This paper also compares the existing methods, namely random and grid search. The experiment resulted in the proposed model being superior to the existing methods. EX3 is better than other experiments and a larger number of hyperparameters and layer-specific hyperparameter values are unimportant.

*This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.*



## Corresponding Author:

Edi Noersasongko

Department Computer science, Dian Nuswantoro University Semarang

Semarang, Indonesia

Email: edi-nur@dosen.dinus.ac.id

## 1. INTRODUCTION

The challenge of finding suitable hyperparameters for convolution neural network (CNN) is still a concern for researchers. CNN has been widely used in computer vision, especially image recognition. In the recognition process, CNN has several advantages, such as automatically extracting important features from each image as well as, saving memory and complexity. The number of hyperparameters strongly influences the CNN performance process. However, hyperparameter optimization is not easy because the accuracy depends on the hyperparameters' quality. Furthermore, there is no standard rule in hyperparameter optimization because each hyperparameter has certain characteristics or is still local optimum. Therefore, the task of hyperparameter optimization is challenging. It is difficult to improve performance with a large number of hyperparameters, the weakness of deep neural networks [1]. So, hyperparameter optimization is a solution [2].

Hyperparameter optimization techniques have been proposed. Grid search (GS) [3] and random search (RS) [4] are popular optimization algorithms. However, the two algorithms do not have a precise learning mechanism to produce an optimal solution, and each solution produced is independent. This mechanism is

fundamental as part of the search activity, which ensures that the optimal solution is global or not trapped in the local optimal in the high dimension of the search space.

Evolutionary algorithm (EA) provides this ability. EA or Neuroevolution as an optimization tool is, popular and widely used to direct the learning process [5], [6]. Based on previous research, neuroevolution is played for optimizing global hyperparameters or the small number of hyperparameters. Most of the studies related to neuroevolution use genetic algorithm (GA) [7]–[11]. GA has two operators which are crossover and mutation. These two operators are used for exploration and exploitation of search space. This paper proposes optimization of hyperparameter and architecture on CNN by using GA or CNN-GA. CNN is a deep learning model widely applied in various objects because of its high performance such as in computer vision, including gender classification [12], image classification [13], [14], vehicle tracking system [15], and e-detection [16].

Despite its capabilities, CNN still has some challenges. The selection of hyperparameters strongly influences its performance. Empirical studies by [17] with text mining data show the strong influence of the larger number of hyperparameters and values of hyperparameter on CNN performance. However, this claim is not necessarily proven in different data. Meanwhile, many papers on CNN hyperparameter using neuroevolution focus on image classification [9], [10], [18]–[23]. Therefore, we focus on CNN optimization using GA on image recognition.

The paper provides some contributions. First, the CNN-GA optimizes 20 hyperparameters for English handwritten recognition. To our knowledge, no research has optimized all hyperparameters (Global, architecture, and layer). Previous research only focused on global parameters or layers. Max-norm weight constraints in convolutional and dense layers also need to be optimized because values significantly impact the search process, and different values affect model performance. Second, this paper designed seven experiments based on the different number of hyperparameters and the different number of optimized hyperparameters. Based on the study of [17], the larger the number of hyperparameters tends to produce the best model performance.

## 2. METHOD

The flow chart for the proposed method is shown in Figure 1. The first stage is pre-processing data. This research used English handwritten recognition (HR) obtained from the National Institute of Standards and Technology (NIST) [24]. The number of datasets was 372.450 records. The details are shown in Figure 2. This figure shows that the distribution of each alphabet digit number is unbalanced, so it becomes an imbalance dataset. Therefore, this research uses the under-sampling approach to balance the datasets. The balance dataset is shown in Figure 3. Next, the image size is reshaped and converted into grayscale, then divided or split into training data and testing data with a proportion of 80:20.

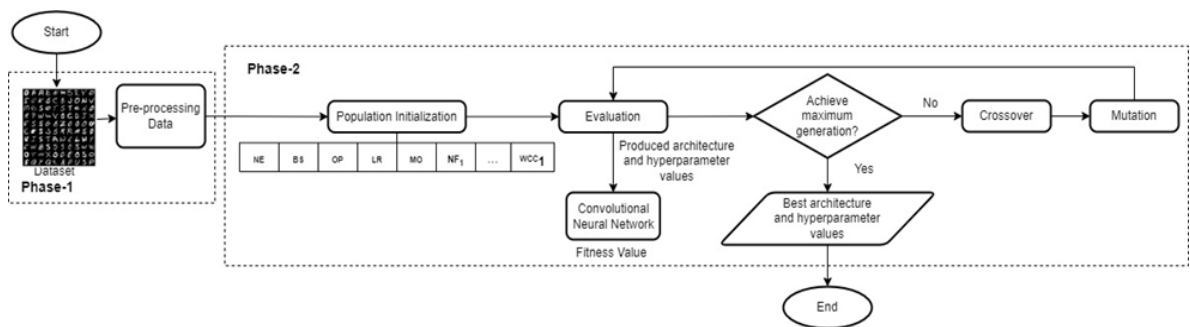


Figure 1. Workflow of the proposed GA-CNN algorithm

We operated standard GA for hyperparameter optimization [25], in which there were five processes: population initialization, evaluation, selection, crossover, and mutation. The initialization of the population used is similar to the study of [17]. However, we eliminated embedding in this process. The initialization of the population is shown in Figure 4. Apart from the same dense layers number (ND) and convolutional layers number (NC), each individual has the same chromosome length based on the number of hyperparameters optimized for each experiment. The list, range, and default of hyperparameter values are presented in Table 1. We refer to the study of [17] that the number of optimized hyperparameters affects the model's performance. Therefore, we conducted seven experiments as done by the study of [17]. This study generated genes for all hyperparameter layers for each layer up to a maximum of NC and ND. This generation process was carried out

to facilitate crossover and mutation. As a result, the number of genes in population size (PS) is a maximum of NC1. This number is based on the max-pooling layer following every convolution layer except the last layer.

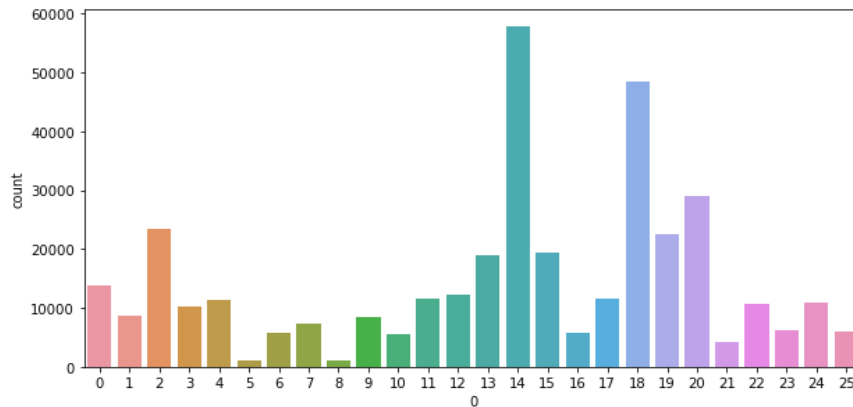


Figure 2. The imbalance of A-Z English HR datasets

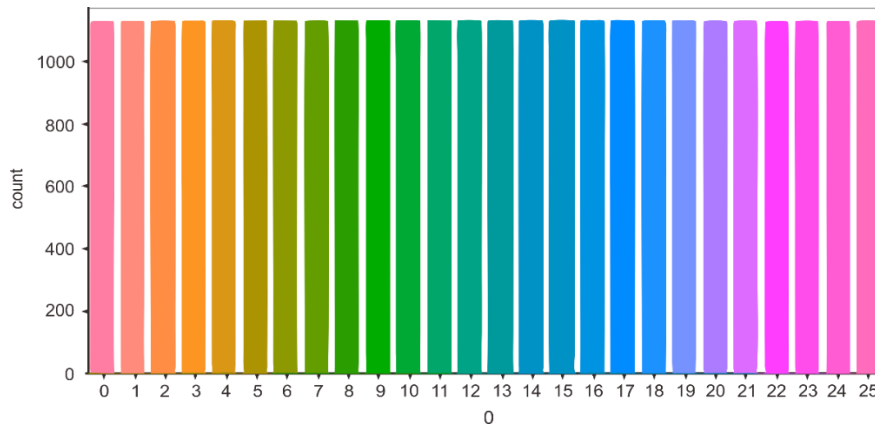


Figure 3. The balance of A-Z English HR datasets

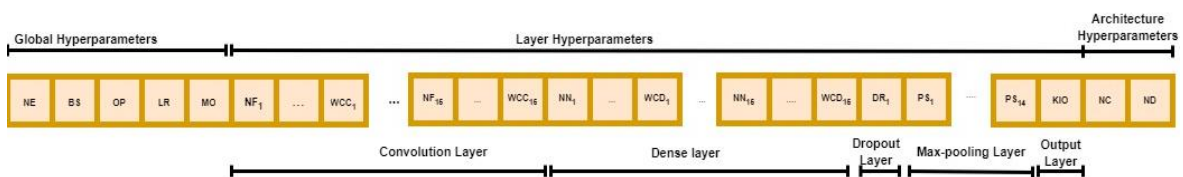


Figure 4. Population initialization of the CNN-GA

After the initialization process, the evaluation process was based on the fitness function, namely the Confusion matrix (accuracy, precision, F1-score, and recall). Figure 1 shows the evaluation process of transferring the architecture and the resulting hyperparameter values to CNN. Next, CNN returns the fitness value. The higher the fitness value, the greater the chance to survive and be selected for the next generation.

Furthermore, the proposed model uses three operators: crossover, mutation, and selection. This research used a uniform mutation type and randomly selected three types of crossover (one-point, two-point, and uniform). NC and ND produced a mated architecture using a one-point crossover operator with a global max-pooling layer. These layers serve as intersection points, which mutate dense layers, or add or remove convolution layers. The selected individual is the best from each generation, which was then selected using elitism.

Table 1. Hyperparameter range and value

Hyperparameter	Description	Range/Values	
Global	NE	The Number of epochs $\alpha:1, \beta:100, \epsilon: 10$	
	BS	Batch Size $\alpha:32, \beta::256, \epsilon: 32$	
	OP	Optimizer [‘Nadam’, ‘Adagrad’, ‘Adadelta’, ‘Sgd’, ‘Rmsprop’, ‘adam’, ‘Adamax’] $\epsilon: ‘Adam’$	
	LR	Learning Rate $\alpha:1e-4, \beta : 1e-2, \epsilon: 1e-4$	
	MO	Momentum $\alpha: 0.0, \beta: 1.0, \epsilon: 0.9$	
Layer	NF	Number of filters (Convolution) $\alpha:32, \beta:512, \epsilon: 64$	
	KS	Kernel Size (Convolution) $\alpha: 1, \beta: 5, \epsilon: 3$	
	AFC	Activation function (Convolutional) [‘Sf’, ‘El’, ‘SL’, ‘SF’, ‘SG’, ‘Tn’, ‘Sid’, ‘HsG’, ‘Linear’], $\epsilon: \epsilon: ‘RL’$	
	KIC	Kernel initializer (Convolution) [‘Zeros’, ‘Glorot_Uniform’, ‘Ones’, ‘Uniform’, ‘Normal’, ‘Glorot_Normal’, ‘Lecun_Normal’, ‘Lecun_Uniform’, ‘He_Normal’, ‘He_Uniform’] $\epsilon: \epsilon: ‘Glorot_Uniform’$	
	WC	Max-norm weight constraint (Convolutional) $\alpha: 1, \beta: 5, \epsilon: 3$	
	NN	Number of neuron (Dense) $\alpha: 1, \beta: 5, \epsilon: 1$	
	AFD	Activation function (Dense) [‘RL’, ‘Sf’, ‘Elu’, ‘SL’, ‘SF’, ‘SG’, ‘Tn’, ‘Sid’, ‘HsG’, ‘Linear’], $\epsilon: ‘RL’$	
	KID	Kernel initializer (Dense) [‘Zeros’, ‘Ones’, ‘Uniform’, ‘Lecun_Normal’, ‘Lecun_Uniform’, ‘Normal’, ‘Glorot_Normal’, ‘Glorot_Uniform’, ‘He_Normal’, ‘He_Uniform’], $\epsilon: ‘Glorot_Uniform’$	
	WC	Max-norm weight constraint (Dense) $\alpha: 1, \beta: 5, \epsilon: 3$	
	DR	Drop rate (Dropout) $\alpha: 0.0, \beta 1.0, \epsilon: 0.2$	
	PS	Pool size (Max-pooling) $\alpha: 2, \beta: 6, \epsilon: 5$	
	KIO	Kernel initializer (Output) [‘Zeros’, ‘Glorot_Normal’, ‘Ones’, ‘Uniform’, ‘Normal’, ‘Glorot_Uniform’, ‘He_Normal’, ‘He_Uniform’,], $\epsilon: ‘Glorot_Uniform’$	
	Architecture	NC	Convolutional layers number $\alpha: 1, \beta: 15, \epsilon: 1$
		ND	Dense layers number $\alpha: 0, \beta: 15, \epsilon: 1$

Note:  $\alpha$ : Min;  $\beta$ :Max,  $\epsilon$ : Default, Sf:softmax,El:Elu,SL: Selu,SF: Softplus, SG: Softsign, Tn:Tanh, Sid: Sigmoid, HsG: Hard\_Sigmoid,RL; Relu

### 3. RESULT AND DISCUSSION

This section discusses the performance of CNN+GA at obtaining the near-optimum combination of the hyperparameters and architecture of CNN. English HR dataset was used to assess the performance of the model [24]. In this study, splits the dataset was split by 80% for training and 20% for testing. The training dataset is also split training dataset and the validation dataset with 80:20 ration [26]. The recognition of English HR is an image recognition that recognises the digits of the alphabet into 28 classes, namely A-Z or a-z. In this experiment, the optimisation method used the training and validation data to produce the near-optimum architecture and combination of hyperparameters. Python was used to implement all hyperparameter optimizations. SciKit-Learn [3] calculated the confusion matrix and visualization tools using Matplotlib [27] and Seaborn. Pandas [28] are used to process datasets, and NumPy [29] handle all scientific computing. Tensorflow [30] and Keras libraries were used to build the CNN model. Distributed evolutionary algorithms in Python (DEAP) was used to create GA. Finally, Google colabPro Plus with GPU high ram was used to perform all the experiments in Table 2.

The use of different hyperparameter values in each layer is marked with an asterisk (\*). We refer to [17] to perform seven experiments based on the different number of hyperparameters and the different number of optimized hyperparameters as presented in Table 2. In [17], the optimization of the number of hyperparameters depends on whether each layer will have the same or different values. This is realized by putting one star (\*) on the hyperparameter layer, which means that the hyperparameter is selected once and is the same for all layers. Meanwhile, the hyperparameter layer with two stars (\*\*) means that each layer will have a different hyperparameter value. Experiment pairs 2-3, 4-5, and 6-7 have the same number of hyperparameters, but the number of optimized hyperparameters differs.

Determination of GA parameter values, namely crossover rate (CR) and mutation rate (MR), is essential in the hyperparameter optimization process. This paper determines CR=0.8 and MR = 0.2. This value is determined based on papers [17], which refer to several papers [8], [10]. The number of generations (Ngen) and population (Npop) selected is 25 in this experiment. Generally, a high population size and a large

generation size will result in better performance. However, this selection will take longer. Based on previous research, the selection of small sizes has also been widely used and proven to produce a good performance [8]–[10], [17], [31].

Table 1. Hyperparameter design for each experiment

Description	Experiment						
	1	2	3	4	5	6	
Global	NE	V	V	V	V	V	V
	BS	V	V	V	V	V	V
	OP	V	V	V	V	V	V
	LR	V	V	V	V	V	V
	MO						V
Layer	N F	V*	V**	V*	V**	V*	V**
	K S	V*	V**	V*	V**	V*	V**
	A F C	V*	V**	V*	V**	V*	V**
	K I C			V*	V**	V*	V**
	W C C			V*	V**	V*	V**
	N N	V*	V**	V*	V**	V*	V**
	A F D			V*	V**	V*	V**
	K I D			V*	V**	V*	V**
	W C D			V*	V**	V*	V**
	D R					V	V
	P S					V*	V**
	K I O					V	V
	Architecture	N C	V	V	V	V	V
N D		V	V	V	V	V	V
Hyperparameter numbers	6	10	10	15	15	20	20
Optimized hyperparameters numbers	6	10	66	15	141	20	159

Notes: \*) All layers with the same Value, \*\*) All layers with different values

This study used an evaluation matrix, which is accuracy, in all experiments. Figure 5 shows the minimum and maximum accuracy results in all experiments. This Figure shows that the higher the size of the population (Npop), the accuracy increases. Experiment 3 (EX 3) has superior accuracy than the other experiments. Meanwhile, Figure 6 shows Mean and Standart Deviasion of Accuracy on CNN-GA. Figure 6 is also in line with Figure 5; a higher number of population sizes results in better average performance. Meanwhile, each population in each generation produces almost the same accuracy value. This means that the distribution of accuracy performance results is uniform so that each population produces a small standard deviation value. EX3 excels with other experiments. Figure 7 shows the average distance (AvgDis) and time execution of accuracy in all experiments.

The smaller AvgDis of each population, the better the accuracy performance produced. This figure shows that EX1 and EX3 have a stable AvgDis, but EX3 has better performance than EX1. Then the experiment resulted in better accuracy performance requiring a longer time. This is in accordance with previous studies, showing that to produce optimal hyperparameters using GA requires more time. Therefore EX3 is an experiment with a longer time than other experiments.

As Figures 5-7 show EX3 is the best experiment among other experiments, with an accuracy of 93.77% and a total execution time of 10 hours 22 minutes. These results indicate that the GA's hyperparameter optimization process can produce the best accuracy if the execution time is long. The comparison of the best accuracy and total execution time is shown in Figure 8.

Furthermore, this paper also presents a comparison of CNN+RS as a comparison of the proposed model shown in Figure 9. Similarly, CNN+RS was built in seven experiments with the same list and range of hyperparameters on CNN+GA. This study set iterations = 150 to run CNN+RS. The best accuracy results show that CNN+GA is superior in all experiments from CNN+RS. Meanwhile, the total execution time resulted in CNN+GA being longer in all experiments than CNN+RS. The best comparison of the best accuracy and total execution time of the two models is shown in Figure 8. Besides RS, this study executed GS by optimizing only six hyperparameters from EX1. This choice was made because only EX1 could cover all 525 CNN's evaluated. If all hyperparameters were optimized, it would require at least 1,48,576 CNNs. Meanwhile, GS could not optimize a number of these CNNs, and the characteristics of CNNs are known to be time-consuming [32]. Therefore, this study used the list and range hyperparameters in GS, similar to the paper [17] presented in Table 3.

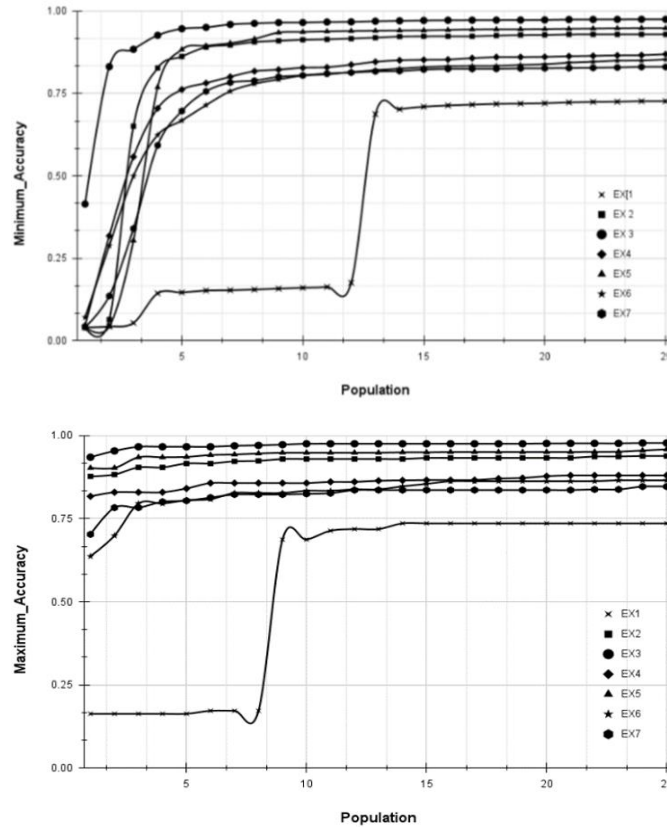


Figure 5. Minimum and maximum of accuracy on CNN-GA

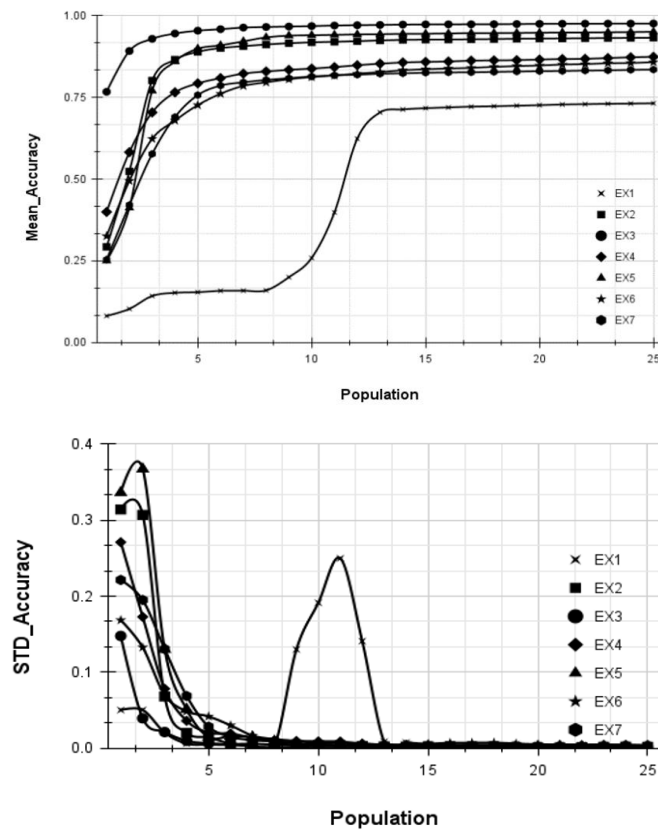


Figure 6. Mean and standart deviation of accuracy on CNN-GA

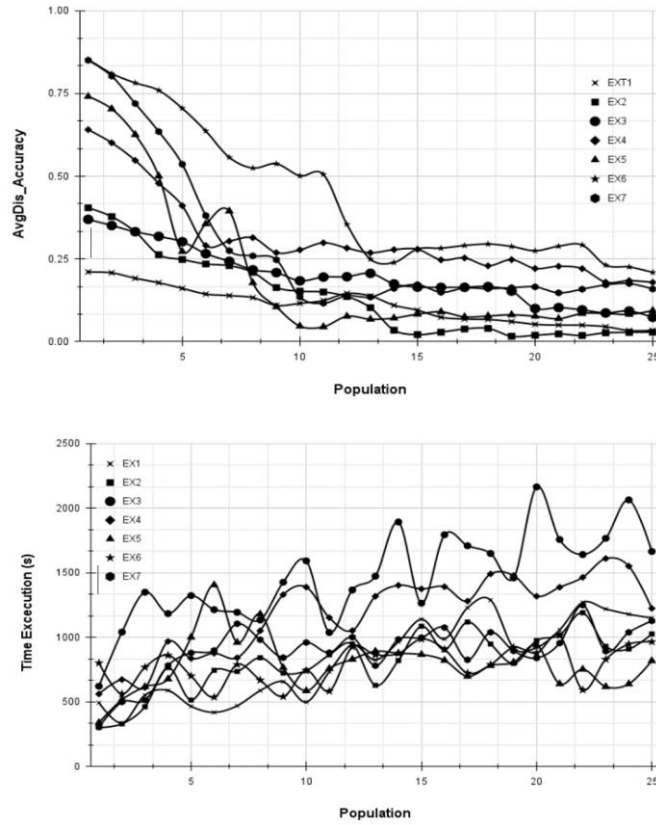


Figure 7. The average distance and time execution of accuracy on CNN-GA

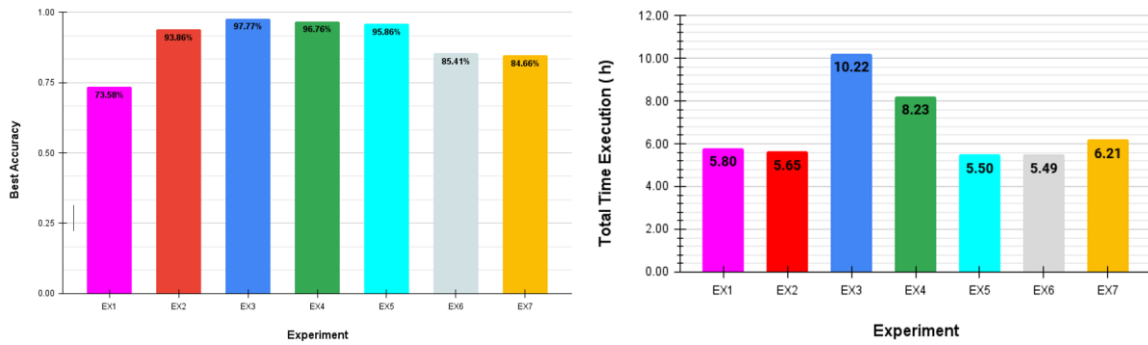


Figure 8. The summarize of best accuracy and total time execution on CNN-GA

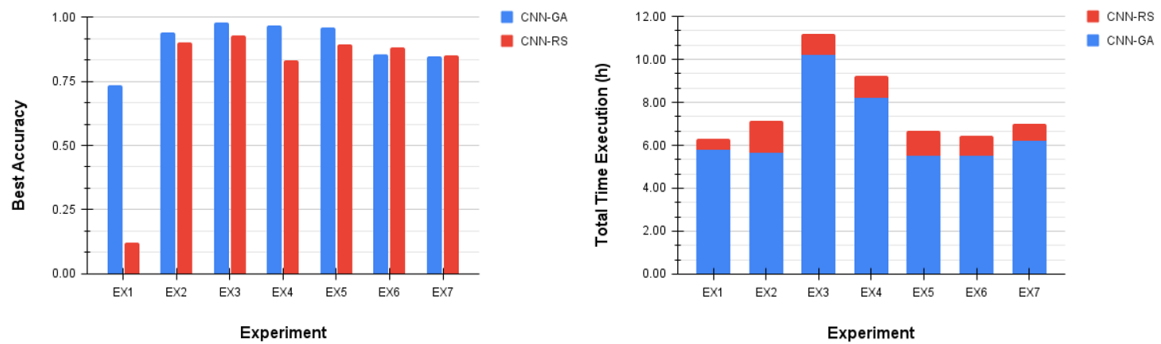


Figure 9. The comparison of CNN+GA and CNN+RS

Table 3. Hyperparameter value sets of GS [17]

Hyperparameter	Set of Values
NE	[20,40,60,80,100]
BS	[64,256]
OP	['Adam', 'Adamax', 'Nadam', 'Adagrad', 'RmsProp']
LR	[1e-3, 1e-2]
NCL	[5, 10, 15]
NDL	[5, 10, 15]

Based on the comparison of the three models, it is found that CNN+GA is superior to CNN+GS and CNN+RS in terms of accuracy performance. The comparison results can be seen in Figure 10. This result is in line with the research obtained by Fatyanosa suggesting that GA is still superior to GS and RS in optimizing CNN hyperparameters with image and text datasets.

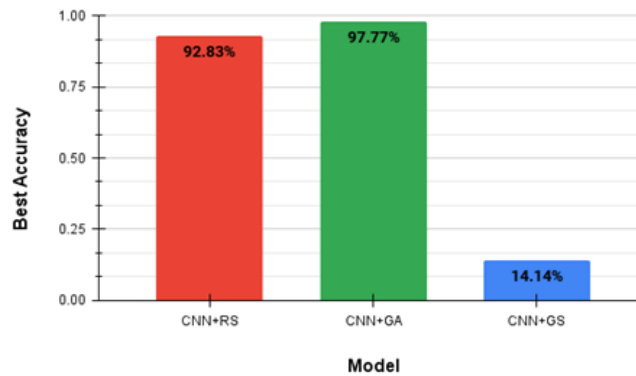


Figure 10. The comparison of three models

The resulting architecture in the best model is shown in Figure 11. The architecture consists of four convolution layers and four dense layers. Deep architecture can study a wide variety of handwritten character datasets and generate architecture for learning handwritten datasets.

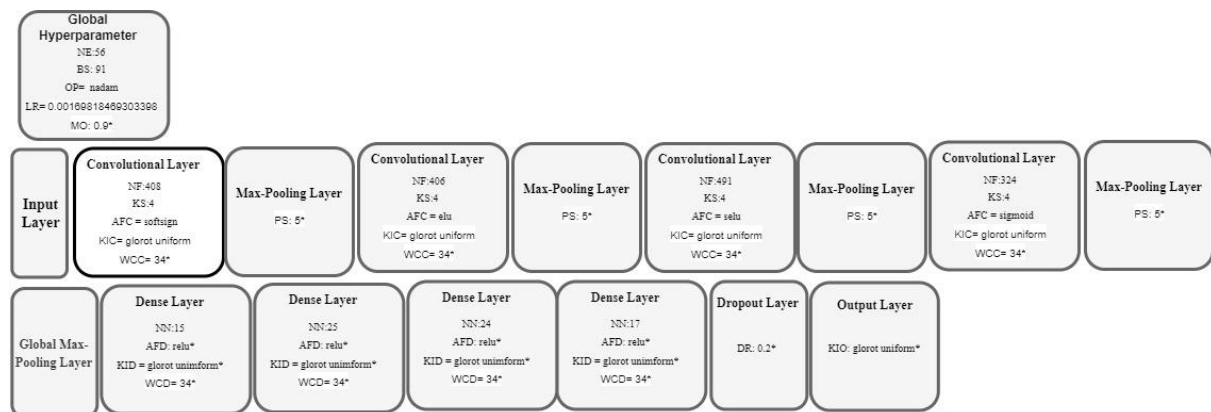


Figure 11. The best individual GA-CNN in EX3

#### 4. CONCLUSION




This paper describes an automated approach to optimize 20 CNN hyperparameters using GA (CNN+GA) on an English handwritten dataset. The results of the seven experiments show that the larger number of hyperparameters and layer-specific hyperparameters are sometimes important as the use of max-norm weight constraint hyperparameters does not significantly impact the search process. It is the standard CNN architecture that produces the best performance compared to CNN with complete hyperparameters. The best result in this study is that EX3 achieves an accuracy of 97.77%.






## REFERENCES

- [1] A. Ghatak, "Deep learning with R," *Deep Learning with R*, pp. 1–245, 2019, doi: 10.1007/978-981-13-5850-0.
- [2] A. Mishra, "Evaluating Machine Learning Models," *Machine Learning in the AWS Cloud*, pp. 115–132, 2019, doi: 10.1002/9781119556749.ch5.
- [3] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [4] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, 2012.
- [5] K. O. Stanley, J. Clune, J. Lehman, and R. Miikkulainen, "Designing neural networks through neuroevolution," *Nature Machine Intelligence*, vol. 1, no. 1, pp. 24–35, 2019, doi: 10.1038/s42256-018-0006-z.
- [6] Z. Fouad, M. Alfonse, M. Roushdy, and A. B. M. Salem, "Hyper-parameter optimization of convolutional neural network based on particle swarm optimization algorithm," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 6, pp. 3377–3384, 2021, doi: 10.11591/eei.v10i6.3257.
- [7] N. Bansal, A. Sharma, and R. K. Singh, "An evolving hybrid deep learning framework for legal document classification," *Ingenierie des Systemes d'Information*, vol. 24, no. 4, pp. 425–431, 2019, doi: 10.18280/isi.240410.
- [8] A. Dahou, M. A. Elaziz, J. Zhou, and S. Xiong, "Arabic Sentiment Classification Using Convolutional Neural Network and Differential Evolution Algorithm," *Computational Intelligence and Neuroscience*, vol. 2019, 2019, doi: 10.1155/2019/2537689.
- [9] T. Hinz, N. Navarro-Guerrero, S. Magg, and S. Wermter, "Speeding up the Hyperparameter Optimization of Deep Convolutional Neural Networks," *International Journal of Computational Intelligence and Applications*, vol. 17, no. 2, 2018, doi: 10.1142/S1469026818500086.
- [10] Y. Sun, B. Xue, M. Zhang, G. G. Yen, and J. Lv, "Automatically Designing CNN Architectures Using the Genetic Algorithm for Image Classification," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3840–3854, 2020, doi: 10.1109/TCYB.2020.2983860.
- [11] S. Sanders and C. Giraud-Carrier, "Informing the use of hyperparameter optimization through metalearning," *Proceedings - IEEE International Conference on Data Mining, ICDM*, vol. 2017-November, pp. 1051–1056, 2017, doi: 10.1109/ICDM.2017.137.
- [12] Y. Nie, W. Ren, B. Liang, J. Dai, and P. Huang, "A study on image based gender classification using convolutional neural network," *ACM International Conference Proceeding Series*, pp. 81–84, 2019, doi: 10.1145/3342999.3343011.
- [13] F. Sultana, A. Sufian, and P. Dutta, "Advancements in image classification using convolutional neural network," *Proceedings - 2018 4th IEEE International Conference on Research in Computational Intelligence and Communication Networks, ICRCICN 2018*, pp. 122–129, 2018, doi: 10.1109/ICRCICN.2018.8718718.
- [14] L. Niharmine, B. Outtaj, and A. Azouaoui, "Tifnagh handwritten character recognition using optimized convolutional neural network," *International Journal of Electrical and Computer Engineering*, vol. 12, no. 4, pp. 4164–4171, 2022, doi: 10.11591/ijece.v12i4.pp4164-4171.
- [15] H. J. Kim, "Multiple vehicle tracking and classification system with a convolutional neural network," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, no. 3, pp. 1603–1614, 2022, doi: 10.1007/s12652-019-01429-5.
- [16] R. Ranjan *et al.*, "A Fast and Accurate System for Face Detection, Identification, and Verification," *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 1, no. 2, pp. 82–96, 2019, doi: 10.1109/tbiom.2019.2908436.
- [17] T. N. Fatyanosa and M. Aritsugi, "Effects of the Number of Hyperparameters on the Performance of GA-CNN," *Proceedings - 2020 IEEE/ACM International Conference on Big Data Computing, Applications and Technologies, BDCAT 2020*, pp. 144–153, 2020, doi: 10.1109/BDCAT50828.2020.00016.
- [18] E. Real *et al.*, "Large-scale evolution of image classifiers," *34th International Conference on Machine Learning, ICML 2017*, vol. 6, pp. 4429–4446, 2017.
- [19] T. Desell, "Developing a volunteer computing project to evolve convolutional neural networks and their hyperparameters," *Proceedings - 13th IEEE International Conference on eScience, eScience 2017*, pp. 19–28, 2017, doi: 10.1109/eScience.2017.14.
- [20] I. Loshchilov and F. Hutter, "CMA-ES for Hyperparameter Optimization of Deep Neural Networks," 2016, [Online]. Available: <http://arxiv.org/abs/1604.07269>.
- [21] A. Martín, R. Lara-Cabrera, F. Fuentes-Hurtado, V. Naranjo, and D. Camacho, "EvoDeep: A new evolutionary approach for automatic Deep Neural Networks parametrisation," *Journal of Parallel and Distributed Computing*, vol. 117, pp. 180–191, 2018, doi: 10.1016/j.jpdc.2017.09.006.
- [22] H. Xie, L. Zhang, and C. P. Lim, "Evolving CNN-LSTM Models for Time Series Prediction Using Enhanced Grey Wolf Optimizer," *IEEE Access*, vol. 8, pp. 161519–161541, 2020, doi: 10.1109/ACCESS.2020.3021527.
- [23] B. Van Stein, H. Wang, and T. Back, "Automatic Configuration of Deep Neural Networks with Parallel Efficient Global Optimization," *Proceedings of the International Joint Conference on Neural Networks*, vol. 2019-July, 2019, doi: 10.1109/IJCNN.2019.8851720.
- [24] P. J. Grother and K. K. Hanaoka, "NIST Special Database 19," pp. 1–30, 2016, [Online]. Available: <https://s3.amazonaws.com/nist-srd/SD19/1stEditionUserGuide.pdf%0Ahttps://www.nist.gov/srd/nist-special-database-19>.
- [25] R. L. Haupt, "An Introduction to Genetic Algorithms for Electromagnetics," *IEEE Antennas and Propagation Magazine*, vol. 37, no. 2, pp. 7–15, 1995, doi: 10.1109/74.382334.
- [26] T. S. Gunawan, A. F. R. M. Noor, and M. Kartiwi, "Development of english handwritten recognition using deep neural network," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 10, no. 2, pp. 562–568, 2018, doi: 10.11591/ijeecs.v10.i2.pp562-568.
- [27] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in Science and Engineering*, vol. 9, no. 3, pp. 90–95, 2007, doi: 10.1109/MCSE.2007.55.
- [28] W. McKinney, "Data Structures for Statistical Computing in Python," *Proceedings of the 9th Python in Science Conference*, pp. 56–61, 2010, doi: 10.25080/majora-92bf1922-00a.
- [29] S. Van Der Walt, S. C. Colbert, and G. Varoquaux, "The NumPy array: A structure for efficient numerical computation," *Computing in Science and Engineering*, vol. 13, no. 2, pp. 22–30, 2011, doi: 10.1109/MCSE.2011.37.
- [30] M. Abadi *et al.*, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems," 2016, [Online]. Available: <http://arxiv.org/abs/1603.04467>.
- [31] G. C. Felbinger, "Optimal CNN Hyperparameters for Object Detection on NAO Robots," 2018.
- [32] C. Fox, "Python for Data Science Primer," pp. 15–25, 2018, doi: 10.1007/978-3-319-72953-4\_2.




**BIOGRAPHIES OF AUTHORS**

**Muhammad Munsarif**    received the Master Degree and Graduated program of computer science, Dian Nuswantoro University Semarang, Indonesia. He is a lecturer in informatics Engineering at Muhammadiyah University, Semarang (UNIMUS). His research interests include computer vision, data science and technopreneurship. He can be contacted at email: [m.munsarif@unimus.ac.id](mailto:m.munsarif@unimus.ac.id)






**Edi Noersasongko**    was born in Semarang, Central Java, Indonesia, in June 1955. He is currently a Senior Professor of artificial intelligence and technopreneurship. He also has a position as the President or Rector of Universitas Dian Nuswanto, Semarang, Indonesia. His area of interests includes artificial intelligence and technopreneurship. He can be contacted at email: [edi-nur@dosen.dinus.ac.id](mailto:edi-nur@dosen.dinus.ac.id)



**Pulung Nurtantio Andono**    was born in Central Java, Indonesia, in September 1982. He received the Bachelor of Engineering from Universitas Trisakti, Jakarta, in 2006, and the Master of Computer Science from Universitas Dian Nuswantoro (UDINUS), in 2009, and the Ph.D. degree from the Institut Teknologi Sepuluh Nopember (ITS), Surabaya, in 2014. He is currently an Associate Professor at Dian Nuswantoro University. His area of interest includes 3D image reconstruction and computer vision. He can be contacted at email: [pulung@dsn.dinus.ac.id](mailto:pulung@dsn.dinus.ac.id)



**Moch Arief Soeleman**    received bachelor and master degree in 1999 and 2004 respectively from Computer Science Department Dian Nuswantoro University, Semarang, Indonesia. Since 2016, he have received Ph.D degree at the Electrical Engineering of Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Indonesia. He works as a lecturer of Computer Science Faculty, Dian Nuswantoro University, Semarang, Indonesia. His research interest includes image processing, computer vision and video processing. He can be contacted at email: [arief22208@gmail.com](mailto:arief22208@gmail.com)