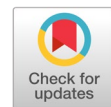


# Improving convolutional neural network based on hyperparameter optimization using variable length genetic algorithm for english digit handwritten recognition



Muhammad Munsarif<sup>a,1</sup>, Edi Noersasongko<sup>b,2,\*</sup>, Pulung Nurtantio Andono<sup>b,3</sup>,  
Mochammad Arief Soeleman<sup>b,4</sup>

<sup>a</sup> Graduated Program of Computer Science, Dian Nuswantoro University Semarang, Indonesia

<sup>b</sup> Departement of Computer Science, Dian Nuswantoro University Semarang, Indonesia

<sup>1</sup> m.munsarif@unimus.ac.id; <sup>2</sup> edi.noer@research.dinus.ac.id; <sup>3</sup> pulung@dsn.dinus.ac.id; <sup>4</sup> arief22208@gmail.com

\* corresponding author

## ARTICLE INFO

### Article history

Received July 30, 2022

Revised December 28, 2022

Accepted January 10, 2023

Available online March 31, 2023

### Keywords

Genetic algorithms ·

Hyperparameter optimization ·

Convolutional neural networks

Handwritten digit recognition

## ABSTRACT

Convolutional Neural Networks (CNNs) perform well compared to other deep learning models in image recognition, especially in handwritten alphabetic numeral datasets. CNN's challenging task is to find an architecture with the right hyperparameters. Usually, this activity is done by trial and error. A genetic algorithm (GA) has been widely used for automatic hyperparameter optimization. However, the original GA with fixed chromosome length allows for suboptimal solution results because CNN has a variable number of hyperparameters depending on the depth of the model. Previous work proposed variable chromosome lengths to overcome the drawbacks of native GA. This paper proposes a variable length GA by adding global hyperparameters, namely optimizer and learning speed, to systematically and automatically tune CNN hyperparameters to improve performance. We optimize seven hyperparameters, such as the learning rate. Optimizer, kernel, filter, activation function, number of layers and pooling. The experimental results show that a population of 25 produces the best fitness value and average fitness. In addition, the comparison results show that the proposed model is superior to the basic model based on accuracy. The experimental results show that the proposed model is about 99.18% higher than the baseline model.



This is an open access article under the [CC-BY-SA](#) license.



## 1. Introduction

Convolutional Neural Networks (CNN) as a powerful technique of deep learning. The accuracy improvement of Convolutional Neural Networks (CNN) is still an interesting study for scholars. CNN is superior performance in computer vision, especially in image recognition [1]–[3]. Furthermore, CNN benefits from high computation [4], a dominant deep learning technique [5], and rich hyperparameter [6], [7], [8] as an advantage of CNN.

The main task of CNN is how to design the best performance of architecture [9]. This task involves validity on hyperparameters, such as the number of filters in layers, how many layers are in the network, the size of the convolution window, the type of optimizer, etc. All of the different hyperparameter combinations create a wide range of possible CNN model architectures [10]. The CNN hyperparameters are divided into three parts, i.e., global hyperparameters, layers, hyperparameters, and architecture hyperparameters. Manually, the testing of various hyperparameters is a good performance. This approach can be carried out on all parts of the hyperparameter. However, the manual approach requires a certain

level of knowledge in deep learning [7], the process of trial and error is tedious and long-time [11]. The solution to this problem is a hyperparameter automated approach.

Another way for hyperparameter optimization is done automatically. Genetic Algorithms (GA) as Evolutionary Algorithms (EA) is widely used for automatically hyperparameter optimization. GA is a search algorithm with a wide range of intervals [12], flexibility to be hybridized with deep learning models, and an extensive library. However, the convergence process of GA failed to obtain global optimal solutions. This shortcoming of GA is solved by [13] with a variable length of the chromosome. This proposed method is proven successfully to overcome premature convergence or local optimal solutions. However, this study only focuses on two hyperparameters (architecture and layers), while other hyperparameters (global) did not use yet. Even though this hyperparameter affects the convergence of CNN model, especially on the chromosome optimizer. Therefore, this paper proposes all GA variable lengths to optimize hyperparameters more efficiently on CNN. On the other hand, the validation of the proposed model uses English Handwritten (EH). This dataset has various types where the writing can be recognized based on the author's writing's thickness, size, shape, and slope. For example, the characters A-Z, a-z, and 0-9 are alphabetic writings in the form of EH. There will be more types because writers have different writing styles, so it is the best way to do recognition.

Several recent studies have explored the use of metaheuristics to optimize CNN hyperparameters in image recognition. In [14] used GA to optimize layer hyperparameters such as kernel size, padding, and activation with Cifar-10 and Cifar-100 datasets. They report that the proposed model performs better than manually hyperparameter optimization. Layer optimization and hyperparameter architecture were carried out by [15] with an accuracy test reaching 95% on the Cifar-10 dataset. Meanwhile, in [16] for Cifar-10 and in [17] for MNIST used GA [18] to optimize all parts of the CNN hyperparameter (layer, architecture, global). They argued that optimizing the learning rate on optimizer can compare the quality of individually produced. In [19], [17] used the Caltech-256 dataset for model validation. They optimize hyperparameters on kernel size and the number of kernels to maintain model depth for good performance. They used the general binary crossover operator of original GA [20]. This implies that hyperparameter values survive for the next generation but are not always successful sequences of consecutive values.

A novel sequential model of crossover operator based on an incremental selective pressure was proposed by [21] to overcome the schedule over evolution in GA. They used a supervised (CIFAR10, MNIST, and Caltech256) dataset to validate the model, with better results for accuracy tests on different data. GA-based hyperparameter optimization layer using Cifar-10 data set achieves 97% precision [22]. Optimization of global hyperparameter and layer hyperparameter based on GA is proposed by [23]. They stated that the dropout value is not significant because this value does not affect the accuracy of the 250 MNIST training data. Three part of hyperparameter optimization on CNN based on GA using a facial emotion recognition dataset was studied by [24]. They reported that the proposed model improved performance by 8% (from 74% to 82%) with genetic algorithms compared to a previous work that utilized a trial and error method. GA provides heuristic-based guided navigation that improves both exploration and exploitation of the solution space, and importance sampling based on Monte Carlo ensures important samples are sampled more frequently for training, as demonstrated by [25] using the MNIST and Cifar-10 datasets to optimize the CNN hyperparameter. They reported that the proposed model provided a winning combination for enhancing the quality of the trained model. CNN Hyperparameter optimization based on GA with MNIST data achieved 99.72% accuracy [26].

GA-based hyperparameter CNN have been extensively worked on and proven to be good. In [13] found the shortcomings of GA, i.e. premature convergence, which causes a local optimum solution. Based on this problem, they proposed a variable length chromosome in GA. The model test results with Cifar-10 data improve the previous work from 51.90% to 88.92%, and the computation is faster. They only use two parts of hyperparameters, i.e. layers and architecture, so if they add one part, i.e. global hyperparameters, it has the potential to improve accuracy. Furthermore, model validation usually uses supervised datasets, i.e. Cifar-10, Cifar-100, MNIST, and Caltech-256. The unsupervised datasets is

needed to enrich the study of GA-based hyperparameter CNN. Therefore, we propose a GA-based hyperparameter CNN with a variable length chromosome with an unsupervised dataset, i.e. EH.

Furthermore, the article is discussed with the following details: Related works are presented in section 2. Section 3 discusses the details of the proposed algorithm, and 4 discusses the experimental results and their analysis. Furthermore, section 5 discusses the conclusions.

## 2. Method

This work involves three phases, as shown in Fig. 1. First, the dataset is preprocessed, the ranges and lists of all the optimized hyperparameters are determined. Then, through the CNN+GA algorithm, the best combination of the CNN hyperparameters is concluded, and finally, using the best combination, the CNN model is built and verified using the test data.

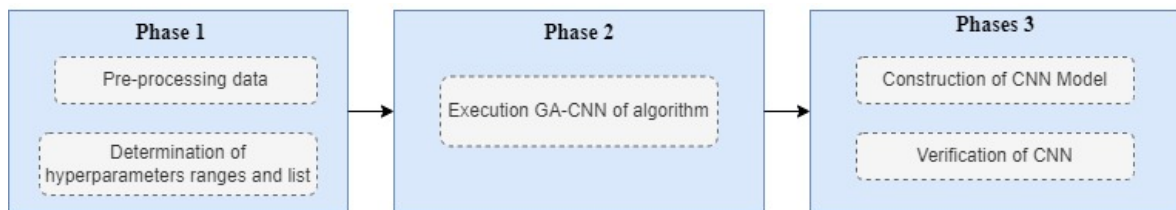


Fig. 1. Flowchart of CNN- GA.

### 2.1. Phase 1

#### 2.1.1. Pre-processing datasets

This paper use English HR datasets that is taken from NIST [27]. The number of datasets is 372.450 records, details as shown in Fig. 2.

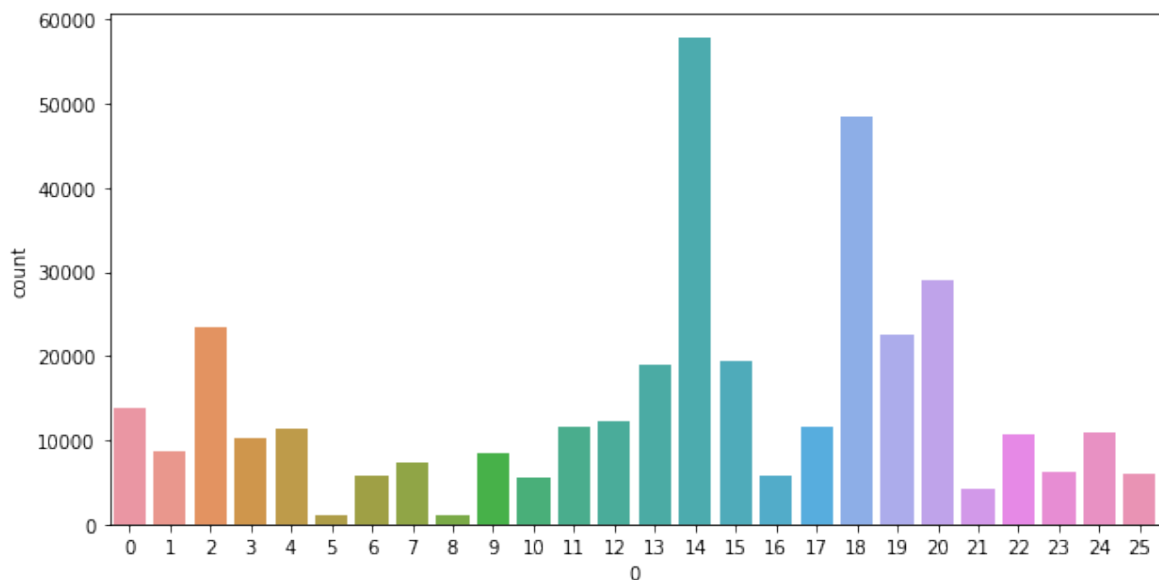


Fig. 2. The imbalance of A-Z English HR datasets

Fig. 2 shows that the dataset distribution is uneven or skewed class distribution. The under-sampling technique is applied to balance by reducing the size of the abundant class. This technique is used because the number of English HR datasets is sufficient. By keeping all samples in the rare class and randomly selecting the same number of samples in the abundant class, a new, balanced dataset can be retrieved for further modelling. The balance dataset is shown in Fig. 3.

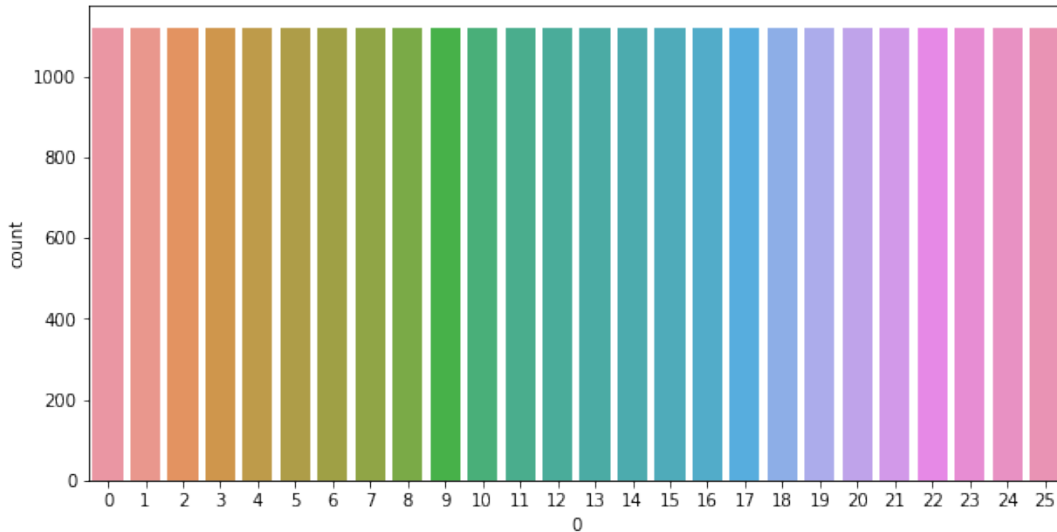


Fig. 3. The balance of A-Z English HR datasets

Next, the image size is reshaped and converted into grayscale (Fig. 4), then divided or split into training data and testing data with a proportion of 80:20.



Fig. 4. Grayscale of A-Z English HR datasets

2.1.2. Determination of hyperparameters ranges and list

This paper optimizes seven hyperparameters divided into three types: global, architecture, and layer. Global hyperparameters can affect the overall model. Parameters included in this category are optimizer and learning rate. Meanwhile, the Hyperparameter layer includes the number of output channels, kernel size, and activation function. Finally, the architecture hyperparameters include the number of the convolutional layer. Range hyperparameters are defined as the depth of the CNN model shown in Table 1.

Table 1. Range of hyperparameter

Hyperparameter	Choices
Number of output	8, 16, 32, 64, 128, 256, 512
Convolutional Filter Size	1x1, 3x3, 5x5, 7x7, 9x9
Activation Function type	ReLu, Tanh, ELU, SELU
Pooling Type	MaxPooling, AVERAGEPooling
Skip Connection	Yes, No
Batch Normalization	Yes, No
Numbers of Layers	≥ 2
Optimizer	Adam, Adamax, Adagrad, Adadelata, Nadam, SGD, RMSprop
Learning rate	[0.001, 0.01]

## 2.2. Phase 2

This section includes the execution GA-CNN of algorithm. The flow diagram of proposed model is illustrated in Fig. 3. The primary process of the proposed model consists of Variable Length Genetic Algorithms, Encoding Scheme, and Fitness of individuals. The tuning of the hyperparameter process at this stage is described in Table 2. The output of this stage is the best combination of hyperparameters from the proposed model.

Table 2. Parameter Tuning

Parameter	value
Crossover Rate	0.8
Mutation Rate	0.2
FitSurvivalRate	0.01
UnFitSurvivalRate	1.0
NGen	10
PopSize	15
n_pahse	15

### 2.2.1. Variable length genetic algorithms

Original GA requires a fixed chromosome length if applied to optimize CNN hyperparameters. This is because CNN has a varying number of convolution layers and has different depths. The variable length of chromosome in GA contains the parameters that represents the solution from the hyperparameter CNN configuration. The solution is encoded in the form of a chromosome discussed in section 2.2.2. Fig 5 describes logical flow for the variable length GA+CNN.

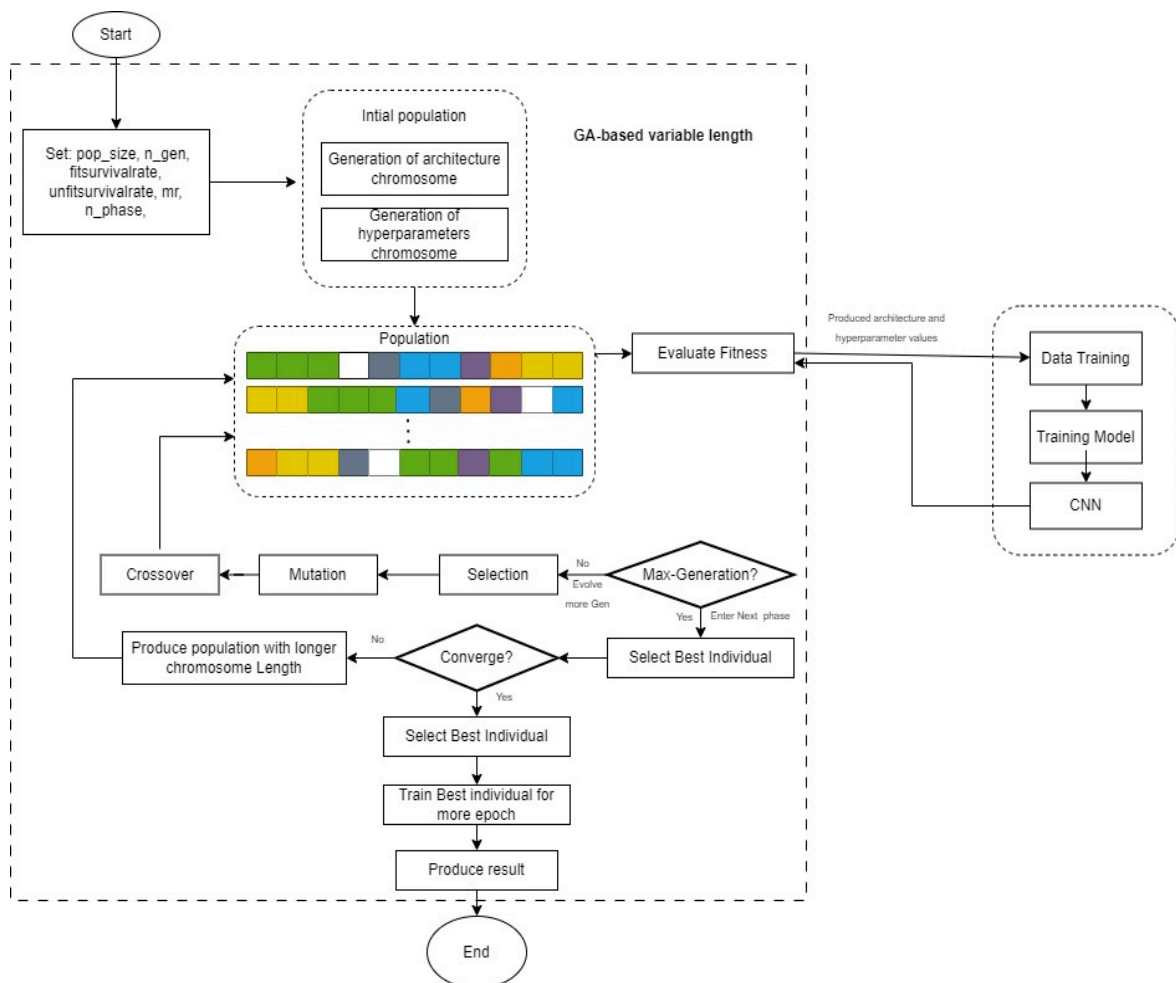


Fig. 5. Logical flow of the GA-CNN algorithm using variable length

Two convolution layers are generated in the initial population, and the individual initializations or solutions to the hyperparameters are randomly generated. Furthermore, individuals are evaluated and sorted based on the best fitness value. The fitness value for the individual is generated from the accuracy value on CNN based on dataset validation. Individuals with the best fitness values are selected and survive to the next generation. The rest, the next generation, is produced by individuals based on the crossover and mutation process using the original GA operator. Crossover makes two individuals parents, resulting in children inheriting the parents' properties. In mutation, some of the individual hyperparameters are changed. After the two layers evolve, the algorithm enters a phase where it is possible to generate generations with more layers and hyperparameters.

**2.2.2. Encoding Scheme**

Chromosomes contain information on each individual. Usually, chromosomes are represented in the form of binary numbers (0 and 1). In optimizing the CNN hyperparameter, the chromosomes in this paper contain information with different values. These values are defined as CNN hyperparameters (global, architecture, and layers) such as optimizer, learning rate, number of layer convolutions, the number of dense layers, kernel size, filter, activation function, etc. Finally, a chromosome is obtained, CNN model is built.

The initial phase of the proposed model, there are two convolutions; layer a and layer b. The suitable hyperparameters are encoded in chromosomes, as illustrated in Fig. 6. The suitable hyperparameters for each convolution layer include feature maps. Furthermore, there are three additional hyperparameters for the two-layer convolution block, including the pooling type and whether or not to include a skip connection. Fig. 7 is an example of the skip connection. If a connection skip exists, the connection layer will skip 1 x 1 convolution, and its output is added to the output of the entire block. In the chromosome, there is an additional "Activation type," which is defined based on the type of activation function that will be used for the whole model. Fig. 6 illustrates the encoding scheme at phase 0.

No. of Output Channels for layer a	Conv. kernel size for layer a	Activation type	Include pool?	Pool Type	Include BN layer a?	No. of Output Channels For layer b	Conv. kernel size for layer b	Include BN for Layer b?	Include skip?
------------------------------------	-------------------------------	-----------------	---------------	-----------	---------------------	------------------------------------	-------------------------------	-------------------------	---------------

Fig. 6. The initial chromosome (Phase 0)

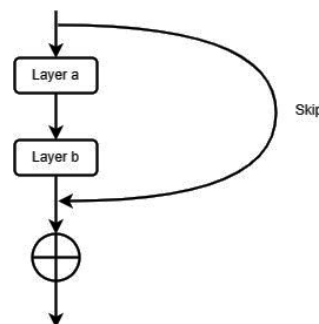


Fig. 7.. Skip connection sample [13]

After phase 0, layer a and layer b can still be added to a convolution layer in the next phase. However, this algorithm added rules to decide whether to add one or two convolution layers in the next phase. Chromosome codification after phase 0 can be illustrated in Fig. 8. New hyperparameters are inserted into the chromosome in the next phase, where the codification is similar to phase 0. Except the hyperparameter does not have "Activation Type" and has the new hyperparameter "Include Layer?" which is 0 or 1. If the new hyperparameter is 1, there will be two layers in this phase, and if 0, there will be only one.

Chromo. From the Previous Phase	No. of Output Channels for layer a	Conv. kernel size for layer a	Include pool?	Pool Type	Include BN layer a?	Include layer b?	No. of Output Channels For layer b	Conv. kernel size for layer b	Include BN for Layer b?	Include skip?
---------------------------------	------------------------------------	-------------------------------	---------------	-----------	---------------------	------------------	------------------------------------	-------------------------------	-------------------------	---------------

Fig. 8. Encoding for Phases after Phase 0.

In the next phase, Chromosome with variable length is formed, and the model architecture is depth. As an illustration, it can be seen in Fig. 9.

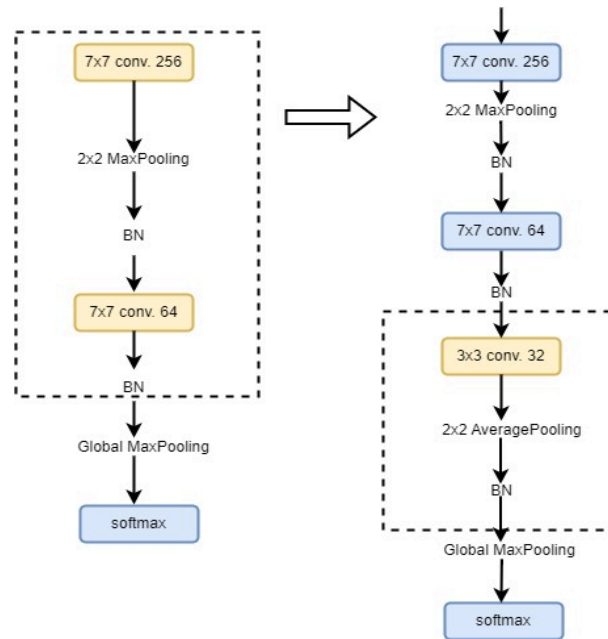


Fig. 9. The model grows deeper when a new phase begins

### 2.2.3. Fitness of individuals

The best individual has the best fitness value accuracy in the validation dataset. One way is to train the model towards convergence and compare it. Although this method is accurate when comparing models on validation datasets, this process is time-consuming and high computational. This problem can be solved by training the model using multiple epochs to compare their relative fitness with each other. A better model in the training process at each epoch tends to have better performance. The number of epochs used in this paper is 5.

### 2.3. Phase 3

This section uses the best hyperparameters and architecture for the training model. In details of this phase are illustrated in Fig. 10. After building the model using training and validation data, the model is verified using data testing.

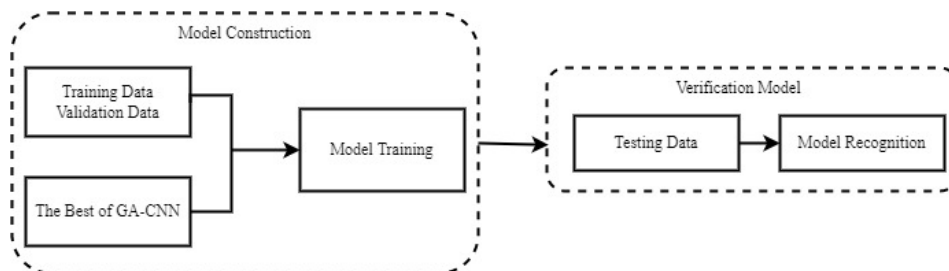


Fig. 10. Phase 3

### 2.4. Evaluation Metrics

To evaluate each individual, we use the accuracy score as a function of fitness formulated with Eq. (1)

$$Accuracy_{score} = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

where TP: True Positif; TN: True Negative; FP: False Positive; FN: False Negative.

## 2.5. Experiment search space and setup

### 2.5.1. Search Space

The search space in GA serves to generate possible solutions [29]. For example, a small search space requires a short time to generate the CNN architecture model, but the resulting model is not necessarily optimal. On the other hand, a vast search space makes it possible to find more solutions so that the resulting CNN model is more optimal but requires more computational power. The search space is detailed in Table 1.

In this paper, we refine the search space used by [13] adding global hyperparameters, namely optimizer and learning rate. Based on previous research, global hyperparameters can maintain the convergence of the model. Furthermore, the search space can expand as the number of layers increases. Initially, the model has two layers, and the search space contains 156,800 combinations of hyperparameters [13]. If the number of layers increases to 10, then the search space grows to about  $10^{25}$ .

### 2.5.2. Experimental Setup

This research was run on colab.google.com with GPU. In order to save time, each individual's fitness is used as an accuracy test after 5 training epochs. Longer chromosomes built on the best individuals in the previous stage indicate the algorithm will enter a new phase where the old best model's trained weights are transferred into the new models, and the new ones are trained for another five epochs to get their fitness value (Fig. 11).

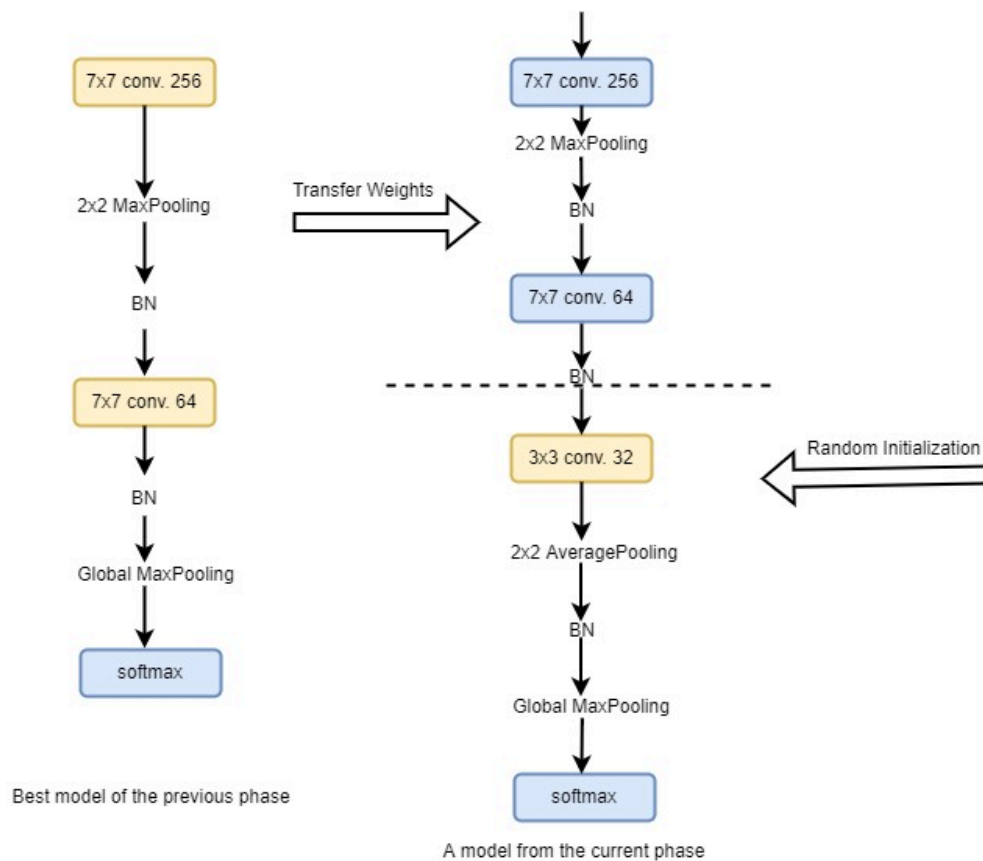


Fig. 11. Design of Weight initialization scheme for deeper models

## 3. Results and Discussion

In this paper, the population is configured as 15 with 8 generations. The best and average fitness results for each generation are shown in Fig. 12. From these results, The best individual is produced by the 8th individual in the 8th generation with a fitness value or accuracy of 99.18%.



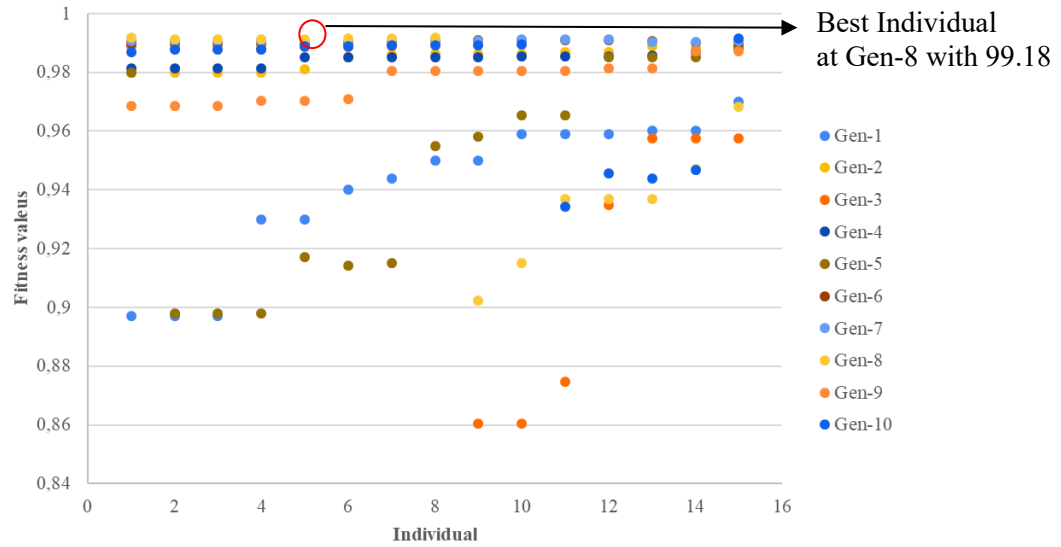


Fig. 12. The Best fitness and average fitness based on the configuration number of population

The details of the hyperparameters in the best population are the first number of output channels is 7x7 with kernel size is 256, the activation function is ReLU, the following number of output channels is 3x3 with kernel size is 64, the optimizer is Adamax with learning rate is 0.001. Adamax is an optimizer which is a variant of the Adam optimizer. However, in practice there is the addition of the infinity norm [28]. The CNN best model visualization is shown in the Fig. 13.

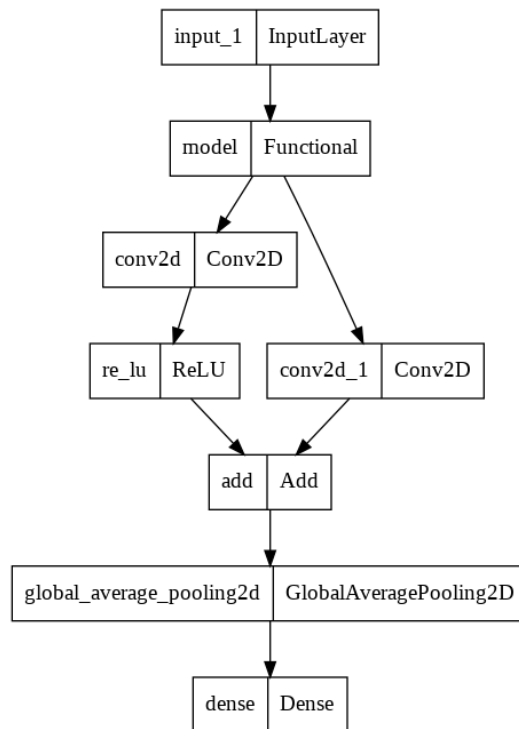


Fig. 13. The best architecture of CNN model

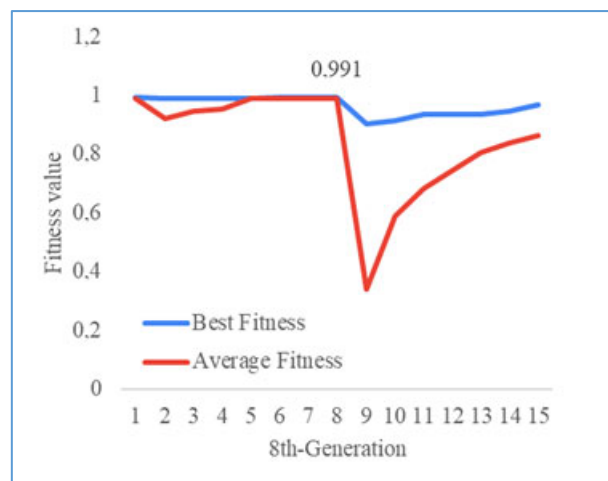
Generation 1 generated a fitness value of 73.86% by applying 0.8 for the crossover operator rate and 0.2 for the mutation operator. The best individuals achieve a fitness value of 85% in this generation. After passing the selected offspring to the next generation, the individuals split or recombine the multiparent and elitism concepts to produce offspring. Big changes in fitness values occur in this phase. The average individual in generation 2 increases dramatically compared to generation 1.

The 2 operators positively impact the fitness values of the individuals. However, there was a slight decrease in generations 3 and 5. Individuals in generation 8 were physically fitter than those in other generations (Table 3).

**Table 3.** The mean of fitness values of individuals based Generation

Generation	Mean Fitness values of all individuals Generated from respective generations	Mean Fitness values of selected individuals
Gen-1	73.86	85.00
Gen-2	98.02	98.62
Gen-3	85.76	91.42
Gen-4	97.82	98.31
Gen-5	77.235	92.77
Gen-6	97.60	98.79
Gen-7	98.30	98.86
Gen-8	98.13	99.07
Gen-9	95.22	94.67
Gen-10	90.65	98.49

Fig. 14 shows the best fitness value and average fitness for each individual in generation of 8. The best fitness and the average fitness for individuals increase. The factor of adding global hyperparameters, namely the optimizer and learning rate, results in a good performance by reducing overfitting by maintaining convergence in the model.



**Fig. 14.** The fitness value of individuals

Furthermore, the proposed method is compared with the baseline model, namely ANN, DNN, CNN, LSTM, biLSTM, GRU, CNN-LSTM, and CNN-RNN. The comparison result of accuracy is shown in Table 4. The proposed model is superior accuracy compared to the baseline model. It means that the hyperparameter approach to the GA-based CNN model is successful.

**Table 4.** The accuracy comparison of proposed models and baseline models

Model	Accuracy
ANN	96.17
DNN	69.33
CNN	98.33
LSTM	96.77
biLSTM	97.49
GRU	97.24
CNN-LSTM	97.84
CNN-RNN	98.25
<b>Proposed Model</b>	<b>99.18</b>

#### 4. Conclusion

This paper developed an existing model about the variable length of GA by adding global hyperparameters, namely optimizer and learning rate, to maintain the convergence of the model. These experiments were performed on an English handwritten digit dataset and compared with the baseline model. The experimental results show that a population of 25 produces the best fitness value and average fitness. In addition, the comparison results show that the proposed model is superior to the baseline model based on accuracy. Furthermore, in future work, we propose hyperparameter values and the number of hyperparameters to be enlarged.

#### Declarations

**Author contribution.** All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

**Funding statement.** None of the authors have received any funding or grants from any institution or funding body for the research.

**Conflict of interest.** The authors declare no conflict of interest.

**Additional information.** No additional information is available for this paper.

#### References

- [1] L. M. Seng, B. B. C. Chiang, Z. A. A. Salam, G. Y. Tan, and H. T. Chai, "MNIST handwritten digit recognition with different CNN architectures," *J. Appl. Technol. Innov.*, vol. 5, no. 1, pp. 7–10, 2021. Available at : [Google Scholar](#).
- [2] S. Ali, Z. Shaukat, M. Azeem, Z. Sakhawat, T. Mahmood, and K. ur Rehman, "An efficient and improved scheme for handwritten digit recognition based on convolutional neural network," *SN Appl. Sci.*, vol. 1, no. 9, pp. 1–9, 2019, doi: [10.1007/s42452-019-1161-5](#).
- [3] M. H. Abed, A. H. I. Al-Rammahi, and M. J. Radif, "Real-Time Color Image Classification Based On Deep Learning Network," *J. Southwest Jiaotong Univ.*, vol. 54, no. 5, 2019, doi: [10.35741/issn.0258-2724.54.5.23](#).
- [4] M. Zohra and D. Rajeswara Rao, "A comprehensive data analysis on handwritten digit recognition using machine learning approach," *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 6, pp. 1449–1453, 2019. doi : [10.17509/ijost.v3i1.10795](#).
- [5] K. Davoudi and P. Thulasiraman, "Evolving convolutional neural network parameters through the genetic algorithm for the breast cancer classification problem," *Simulation*, vol. 97, no. 8, pp. 511–527, 2021, doi: [10.1177/0037549721996031](#).
- [6] A. N. I. Hui, A. B. Huddin, M. F. Ibrahim, F. H. Hashim, and S. A. Samad, "GA-deep neural network optimization for image classification," *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 8, no. 1.6 Special Issue, pp. 238–245, 2019, doi: [10.30534/ijatcse/2019/3681.62019](#).
- [7] T. N. Fatyanosa and M. Aritsugi, "An Automatic Convolutional Neural Network Optimization Using a Diversity-Guided Genetic Algorithm," *IEEE Access*, vol. 9, pp. 91410–91426, 2021, doi: [10.1109/ACCESS.2021.3091729](#).

- [8] I. Priyadarshini and C. Cotton, "A novel LSTM-CNN-grid search-based deep neural network for sentiment analysis," *J. Supercomput.*, vol. 77, no. 12, pp. 13911–13932, 2021, doi: [10.1007/s11227-021-03838-w](https://doi.org/10.1007/s11227-021-03838-w).
- [9] A. Aghaebrahimian and M. Cieliebak, "Hyperparameter tuning for deep learning in natural language processing," *CEUR Workshop Proc.*, vol. 2458, 2019. doi : [10.21256/zhaw-18993](https://doi.org/10.21256/zhaw-18993).
- [10] F. M. Talaat and S. A. Gamel, "RL based hyper-parameters optimization algorithm (ROA) for convolutional neural network," *J. Ambient Intell. Humaniz. Comput.*, 2022, doi: [10.1007/s12652-022-03788-y](https://doi.org/10.1007/s12652-022-03788-y).
- [11] S. Albahli, F. Alhassan, W. Albattah, and R. Ullah, "Handwritten digit recognition: Hyperparameters-based analysis," *Appl. Sci.*, vol. 10, no. 17, 2020, doi: [10.3390/app10175988](https://doi.org/10.3390/app10175988).
- [12] M. A. A. Albadr, S. Tiun, M. Ayob, and F. T. AL-Dhief, "Spoken language identification based on optimised genetic algorithm-extreme learning machine approach," *Int. J. Speech Technol.*, vol. 22, no. 3, pp. 711–727, 2019, doi: [10.1007/s10772-019-09621-w](https://doi.org/10.1007/s10772-019-09621-w).
- [13] X. Xiao, M. Yan, S. Basodi, C. Ji, and Y. Pan, "Efficient Hyperparameter Optimization in Deep Learning Using a Variable Length Genetic Algorithm," 2020. doi : [10.48550/arXiv.2006.12703](https://doi.org/10.48550/arXiv.2006.12703).
- [14] Y. Sun, B. Xue, M. Zhang, G. G. Yen, and J. Lv, "Automatically Designing CNN Architectures Using the Genetic Algorithm for Image Classification," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 3840–3854, 2020, doi: [10.1109/TCYB.2020.2983860](https://doi.org/10.1109/TCYB.2020.2983860).
- [15] P. Nirwan and G. Singh, "Segmentation and identification of bilingual offline handwritten scripts (devanagari and roman)," *Int. J. Recent Technol. Eng.*, vol. 8, no. 2 Special Issue 6, pp. 603–607, 2019, doi: [10.35940/ijrte.B1178.0782S619](https://doi.org/10.35940/ijrte.B1178.0782S619).
- [16] A. J. Reiling, "Convolutional Neural Network Optimization Using Genetic Algorithms," 2017. Available at : [Semantic Scholar](https://www.semanticscholar.org/).
- [17] A. Bhandare and D. Kaur, "Designing convolutional neural network architecture using genetic algorithms," in *2018 World Congress in Computer Science, Computer Engineering and Applied Computing, CSCSE 2018 - Proceedings of the 2018 International Conference on Artificial Intelligence, ICAI 2018*, 2018, pp. 150–156. doi: [10.21307/ijanmc-2021-024](https://doi.org/10.21307/ijanmc-2021-024).
- [18] F. Mattioli, D. Caetano, A. Cardoso, E. Naves, and E. Lamounier, "An experiment on the use of genetic algorithms for topology selection in deep learning," *J. Electr. Comput. Eng.*, vol. 2019, 2019, doi: [10.1155/2019/3217542](https://doi.org/10.1155/2019/3217542).
- [19] S. Loussaief and A. Abdelkrim, "Convolutional Neural Network hyper-parameters optimization based on Genetic Algorithms," *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 10, pp. 252–266, 2018, doi: [10.14569/IJACSA.2018.091031](https://doi.org/10.14569/IJACSA.2018.091031).
- [20] A. Hassanat, K. Almohammadi, E. Alkafaween, E. Abunawas, A. Hammouri, and V. B. S. Prasath, "Choosing mutation and crossover ratios for genetic algorithms-a review with a new dynamic approach," *Inf.*, vol. 10, no. 12, 2019, doi: [10.3390/info10120390](https://doi.org/10.3390/info10120390).
- [21] F. Johnson, A. Valderrama, C. Valle, B. Crawford, R. Soto, and R. Nanculef, "Automating Configuration of Convolutional Neural Network Hyperparameters Using Genetic Algorithm," *IEEE Access*, vol. 8, pp. 156139–156152, 2020, doi: [10.1109/ACCESS.2020.3019245](https://doi.org/10.1109/ACCESS.2020.3019245).
- [22] C. Li *et al.*, "Genetic algorithm based hyper-parameters optimization for transfer convolutional neural network," 2022. doi: [10.1117/12.2637170](https://doi.org/10.1117/12.2637170).
- [23] J. H. Yoo, H. Il Yoon, H. G. Kim, H. S. Yoon, and S. S. Han, "Optimization of Hyper-parameter for CNN Model using Genetic Algorithm," *2019 IEEE Int. Conf. Electr. Control Instrum. Eng. ICECIE 2019 - Proc.*, 2019, doi: [10.1109/ICECIE47765.2019.8974762](https://doi.org/10.1109/ICECIE47765.2019.8974762).
- [24] R. Zatarain Cabada, H. Rodriguez Rangel, M. L. Barron Estrada, and H. M. Cardenas Lopez, "Hyperparameter optimization in CNN for learning-centered emotion recognition for intelligent tutoring systems," *Soft Comput.*, vol. 24, no. 10, pp. 7593–7602, 2020, doi: [10.1007/s00500-019-04387-4](https://doi.org/10.1007/s00500-019-04387-4).
- [25] A. Shrestha and A. Mahmood, "Optimizing deep neural network architecture with enhanced genetic algorithm," in *Proceedings - 18th IEEE International Conference on Machine Learning and Applications, ICMLA 2019*, 2019, pp. 1365–1370. doi: [10.1109/ICMLA.2019.00222](https://doi.org/10.1109/ICMLA.2019.00222).

- 
- [26] A. Baldominos, Y. Saez, and P. Isasi, "Model Selection in Committees of Evolved Convolutional Neural Networks Using Genetic Algorithms," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11314 LNCS, pp. 364–373, 2018, doi: [10.1007/978-3-030-03493-1\\_39](https://doi.org/10.1007/978-3-030-03493-1_39).
- [27] P. J. Grother and K. K. Hanaoka, "NIST Special Database 19," 2016. doi : [10.18434/T4H01C](https://doi.org/10.18434/T4H01C).
- [28] N. K. Manaswi, "Deep Learning with Applications Using Python," *Deep Learning with Applications Using Python*. 2018. doi: [10.1007/978-1-4842-3516-4](https://doi.org/10.1007/978-1-4842-3516-4).
- [29] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: past, present, and future," *Multimed. Tools Appl.*, vol. 80, no. 5, pp. 8091–8126, 2021, doi: [10.1007/s11042-020-10139-6](https://doi.org/10.1007/s11042-020-10139-6).