Journal of Theoretical and Applied Information Technology
15th September 2020. Vol.98. No 17

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

GENETIC ALGORITHM APPROACH FOR AUTOMATED GENERATION OF NEURAL NETWORKS ARCHITECTURE FOR ROBUST DIGIT RECOGNITION

[1]RULLY SOELAIMAN, [2]YOLANDA HERTITA PRATAMA, [3]M.M. IRFAN SUBAKTI, [4]YUDHI PURWANANTO

[1,2,3,4]Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Jawa Timur, Indonesia

E-mail: [1]rully130270@gmail.com, [2]yolandahertita903@gmail.com, [3]yifana@gmail.com, [4]purwananto@gmail.com

ABSTRACT

Digit recognition is a special part of research in Optical Character Recognition. It is a common technique to recognize the numeric characters from printed images. A solution's design for the digit recognition problem on a case study of SPOJ Hard Image Recognition (HIR) has been proposed in this paper. The case study's problem has challenging constraints such as runtime limit and source code limit and it has many noisy images. An artificial neural network has been implemented to solve this problem in consideration of its simplicity yet powerful enough algorithm. The selection of best ANN architecture is commonly achieved through trial and error process, which is a very time-consuming process. This paper also provides the use of a Genetic Algorithm to determine the architecture of ANN automatically. The creation of the dataset also has an important role to improve the accuracy. The proposed architecture development successfully passed the challenging constraints and achieved a high score of 108 at SPOJ HIR. The score obtained by using GA is higher than our predetermined ANN architecture.

Keywords: *Artificial Neural Network, Digit Recognition, Genetic Algorithm, Optimization, Pattern Classification*

1. INTRODUCTION

Optical Character Recognition (OCR) is a field of research in pattern recognition. It refers to the mechanical or electronic translation of images of handwritten, typewritten or printed characters into the machine-editable text. It is a common technique to transform printed images so that they can be electronically edited, stored, searched, and displayed. Digit recognition is a special case for OCR, the system only needs to recognize 10 numeric characters (0-9). The application of digit recognition varies in different fields such as ID card recognition, license plate recognition, postal address interpretation, captcha, etc.

Digit recognition or OCR can be further divided into typewritten or printed character and handwritten character recognition. Recognition of the typewritten characters is easier because they are regular and uniform in shape [1]. The differences between these characters are only its colour, font size, etc. On the other hand, handwritten characters can vary for different handwriting styles. One character written by one person for several times can be different. Despite the recognition of typewritten or printed characters is easier, the problem of its recognition is still an active area of research. The key modules of these existing solutions are preprocessing, character segmentation, and character recognition.

Many sophisticated pattern recognition techniques have been proposed such as Support Vector Machine (SVM), deep learning, and others. In [2], the authors used feature extraction and compared most commonly classifiers, SVM and ANN. Despite the promising result coming from SVM, a simple algorithm is still needed so that it can be applied to the systems that have limited resources.

This paper proposed a digit recognition using artificial neural network (ANN). A neural network is a massively parallel distributed structure made up of simple processing units that can learn and therefore generalize [3]. Though it is extremely difficult for researchers to explain the relationships between the input features and the output, ANN has advantages that are its high tolerance of noisy data, its ability to classify data which it never been trained, and simply in structure.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The case study of Sphere Online Judge (SPOJ) Hard Image Recognition (HIR) [4] will be used to test the performance of the developed algorithm. In this problem, there are a set of 'X' and '.' characters that represent binary images of 6 digits. The program should recognize the 6-digit numbers of every image. The main idea in our approach is training ANN offline to obtain components such as weights and biases then write a test program to be submitted to the online judge. The test program will not perform the learning process, so the same architecture and the extracted components from the training process will be used.

The selection of best ANN architecture is commonly achieved through trial and error process [5], some different architectures are examined and compared to one another. This is a very time-consuming process. However, some rules-of-thumb are available for selecting the number of neurons in the hidden layers. A rough approximation for the number of neurons can be obtained by using the rule of a geometric pyramid proposed by Masters [6]. The number of neurons in consecutive layers has a shape of a pyramid and decreases from the direction of input to the output. The numbers of neurons in consecutive layers are forming a geometric sequence.

But these rules-of-thumb are not considered always to be true. Benardos and Vosniakos [7] suggest a method to optimize the feedforward ANN architecture. The solution space consists of all the different combinations of hidden layers and the number of neurons in the hidden layers. Genetic Algorithm has been employed to search the solution space for the best architectures.

This paper also provides the use of a Genetic Algorithm (GA) to determine the architecture of ANN which are used to classify the digits. The hidden layers of ANN have *n*-nodes which every node has a weight and single bias. The value of *n* in the hidden layer will be optimized by using GA. The objective function used in GA formulated by combining training error and generalization error of the network. This is the function whose value must be minimized by the GA.

This paper is organized as follows. In Section 2, the problem and dataset used for this analysis are introduced. Section 3 explains the algorithms used in our approach. Section 4 talks about our designed method to generate ANN architectures using GA and our approach to solve the problem. In Section 5, the evaluation method is explained, and the results are presented in Section 6. Conclusions are presented in section 7.

2. PROBLEM STATEMENT

In this paper, we have proposed a design solution to the digit recognition problem on a case study of Sphere Online Judge (SPOJ) Hard Image Recognition (HIR) [4]. SPOJ is an online judge, problem set and contest hosting service. This online judge accepts over 40 programming languages. SPOJ consists of five problem categories, there are basic, classical, challenge, partial, and riddle. Hard Image Recognition is a problem from the challenge section. The more our solution gives the right output of test cases, the higher score it can be obtained.

At this problem, there are black and white pictures, consisting of a white background with a black raster image of 6-digit numbers on it. Character ‘X’ denotes the colour white (the background) and character ‘.’ denotes the colour black (the foreground). The numbers can be written in different fonts. In some cases, the image may be blurred, transformed, and can have many undesired noises such as crossed-out characters. The program should recognize the 6-digit numbers of every image.

On the SPOJ HIR open contest forum discussion [8], there are 13 examples of input images in TXT files, which contains the set of characters 'X' and '.'. The example of these files can be seen in Figure 1. The dataset used for the learning process was taken from these 13 sample images.

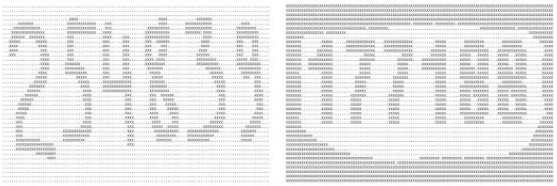


Figure 1: Example of Input Image

From the sample images, it can be seen that the writing style for the digits is likely the typewritten digits. We also generate a dataset to add more variation in data training. The training set was created using fonts that have been downloaded through a website that provides various types of fonts. The dataset consists of digits from 0 to 9 using different fonts and in several different sizes. The example of the created dataset can be seen in Figure 2.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

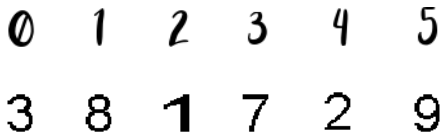


Figure 2: The Created Dataset

3. RELATED DEFINITIONS

3.1 Depth First Search

Depth First Search (DFS) is an algorithm for searching a graph/tree, in a depth-wise manner. Depth-first search explores edges out of the most recently discovered vertex that still has unexplored edges leaving it. After all edges of vertex v have been visited, the search will perform backtracking to explore the edges of the vertex v that have not yet been visited. This process continues until we have discovered all the vertices that are reachable from the original source vertex. If any undiscovered vertices remain, then depth-first search selects one of them as the new source, and it repeats the search from that source. The algorithm repeats this entire process until it has discovered every vertex [9]. The illustration of this algorithm can be seen in Figure 3.

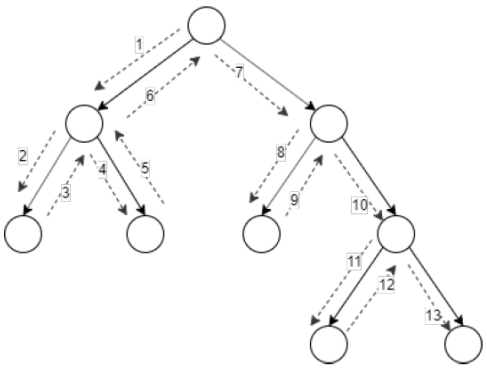


Figure 3: Illustration of Depth First Search

The DFS function has been used to perform segmentation on the input images. This process aims to obtain parts that are the foreground of the image. The results obtained later will be filtered on whether the part is a digit or noise.

3.2 Nearest Neighbour Image Scaling

Nearest neighbour is a simple and fast algorithm for the process of changing the size of an image. Nearest neighbours are not like other algorithms, such as bilinear, bicubic, spline and others, which use interpolation for neighbouring pixels resulting in smoother or better images [10]. Nearest neighbour is the appropriate algorithm to solve the SPOJ Hard Image Recognition (HIR) problem which is a binary image so that there are no new pixel values other than black or white generated in the interpolation process.

In the nearest neighbour technique, the empty box will be filled with neighbouring pixels. The constructed image will be smaller, larger, or equal in size depending on the scaling ratio. This algorithm requires horizontal ratio and vertical ratio on the image to make changes in the size. The horizontal ratio is obtained from the ratio between initial height and final height, while the vertical ratio is obtained from the ratio between the initial width and the final width. The pseudocode of this algorithm is given in Figure 4.

Algorithm 1 Nearest Neighbor Image Scaling

```
1: INTEGER w1, h1, w2, h2
2: ARRAY pixels[w1 * h1], temp[w2 * h2]
3: DOUBLE xratio ← w1/ (double) w2
4: DOUBLE yratio ← h1/ (double) h2
5: DOUBLE px, py
6: for i ← 0, h2 do
7:     for j ← 0, w2 do
8:         px ← [j * xratio]
9:         py ← [i * yratio]
10:        temp[(i * w2) + j] ← pixels[(int)((py * w1) + px)]
11:    end for
12: end for
```

Figure 4: Pseudocode of Nearest Neighbor Image Scaling

3.3 Artificial Neural Network

Artificial Neural Network (ANN) is an information processing system inspired by the workings of the biological nervous system, especially on human brain cells in processing information. The Neural Network consists of a large number of information processing elements (neurons) that are interconnected and work together to solve a particular problem. ANN will carry out the learning process to produce the best weight.

The approach that will be used to solve this SPOJ HIR problem is using the weights of ANN that have been trained to calculate the output of the image. The test program that will be submitted to the online judge will not perform the learning process, so it will be using the same architecture and the extracted components from the training process.

In ANN, each layer has some neurons. The architecture of ANN used in this paper is Single-Layer Feedforward Networks. This neural network has only one layer with a connected weight. The network receives input through the hidden layer. This network has only one input layer and one output layer.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

In mathematical notation, the calculation of the output in neurons *k* can be done by Eq. (1), then after adding bias and activation functions, the final output can be done by Eq. (2). An overview of the ANN algorithm, in general, can be seen in Figure 5.

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \quad (1)$$

$$y_k = \varphi(u_k + b_k) \quad (2)$$

Where $x_1, x_2, ..., x_m$ is the input signal; $w_{k1}, w_{k2}, ..., w_{km}$ is the weight of the neuron k; u_k is the sum of the linear output functions of the input signal; b_k is bias; φ is an activation function, and y_k is the output signal from neurons. The purpose of adding bias b_k is to apply affine transformations to output.

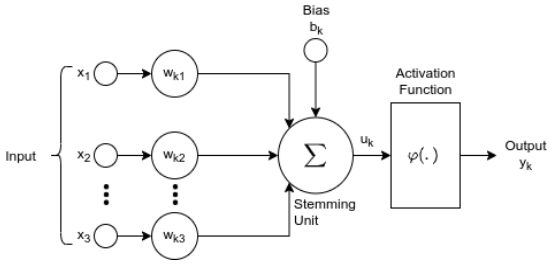


Figure 5: The General Architecture of ANN

The training process in ANN is as follows:

1) Calculate the output
2) Comparing the received output with the desired target
3) Adjust the weights and repeat the process.

ANN is a simple algorithm but it has many advantages: it can quickly classify data which it has never been trained before, its high tolerance of noisy data, and very simple in structure so it can be applied to systems that have limited resources. This is the right choice of the algorithm to solve the SPOJ Hard Image Recognition (HIR) problem that has challenging constraints such as source code limit and runtime limit.

3.4 Genetic Algorithm

Genetic Algorithm (GA) is a search technique in the field of computing that is used to generate solutions to optimization and search problems. GA is a part of the Evolutionary Algorithm, an algorithm that is inspired by the natural evolutionary process where the most superior individuals will survive, while weak individuals will become extinct [11]. GA uses techniques inspired by evolutionary biologies such as inheritance, mutation, selection, and crossover.

The process begins with a set of individuals which is called a population, which has several chromosomes arranged by several genes which are representations of candidate solutions to a problem. The best candidates will be selected through a selection process based on the calculated fitness value for each chromosome in the population. Selected candidates from this process will form the mating pool, a set where two parents will be formed from this process. In Evolutionary Algorithms, the principle of survival arises because of the process of reproduction. The resulting offspring will carry the gene of its parents; therefore, parents are selected from the mating pool which is the collection of the best candidates from a population.

The process of GA is as follows [12]:

1) Generate a random population of chromosomes.
2) If the termination criteria are satisfied, stop. Otherwise, continue with step 3.
3) Determine the fitness of each chromosome.
4) Apply crossover and mutation to selected chromosomes from the current generation to generate a new population of chromosomes for the next generation.
5) Return to step 2.

GA is used to determine the architecture of ANN automatically. The hidden layers of ANN have *n*-nodes which every node has a weight and single bias. The value of *n* in the hidden layer will be optimized by using GA.

4. Proposed Approach

In the problem, there are a set of 'X' and '.' characters that represent binary images of 6 digits. The program should recognize the 6-digit numbers of every image. The problem has constraints as below.

1) The runtime limit or program execution time limit is 0.1 seconds.
2) Memory usage limit is 1536MB.
3) The size limit of the source code sent to the online judge is 0.15MB.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The images may have varying digit sizes, varying types of digit typography, and augmentation from these digits such as rotation. Deterministic method to approach the solution will be prone to error. The limited size of source code, the execution time limit to produce the expected output, and the variations in written digits are the main reasons why a machine-learning approach, Artificial Neural Network algorithm, is used to solve this problem. The general process of solving the SPOJ HIR problem as follows.

1) Prepare dataset using fonts that have been downloaded and images are taken from the SPOJ HIR forum.
2) Optimize the ANN architecture using GA.
3) Train the selected ANN architecture to obtain components such as weights and biases.
4) Write a test program to be submitted to the online judge using extracted weights and biases.

This section divided into several sections that explain the strategy of solving the SPOJ HIR starting from the preprocessing stage, how to use ANN as the main algorithm in solving problems and the preparation of datasets used in training, and how to optimize the architecture of ANN using GA.

4.1 Image Preprocessing

This stage is located in the test program, which aims to extract the digits from the image which contains 6 digits by doing segmentation using Depth First Search algorithm. The example of the extracted digit can be seen in Figure 6.

Figure 6: Example of Segmentation Results in Images

After getting the digit segmented candidates, the process continues with removing noise from digits, and normalizing the size of digits so that all input images will have the same number of features.

4.1.1 Noise removal

In the SPOJ HIR problem, several images have noise such as dot noise, line noise and crossed-out noise. Examples of these noisy images can be seen in Figure 7.

In images that have dot noise and line noise as in the examples of Figure 7a and Figure 7b, the length and width of the segmented area are used to adjust the threshold. If there is an element that is wide or higher than the average, then the element will be removed from the candidate.

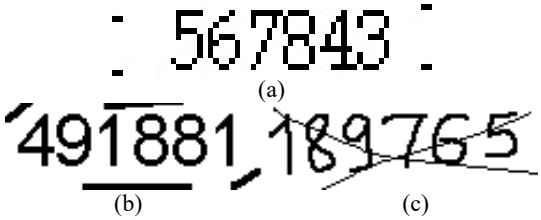


Figure 7: Examples of Noisy Images in HIR Problem: (a) Dot Noise, (b) Line Noise and (c) Crossed-out Noise.

In images that have crossed-out noise as in the examples of Figure 7c, the inpainting technique by using neighbour information from a pixel that contains the 'X' character is used. The information was extracted by using 8-connectivity. This method has been proposed by Axel et al. [13] as one of the features for automatically identifying and removing crossed-out text in off-line handwriting. There is a threshold of how many neighbours that also have 'X' character, if less than the threshold then the pixel of those positions will be transformed from character 'X' into character '.'. The example of this process is shown in Figure 8.

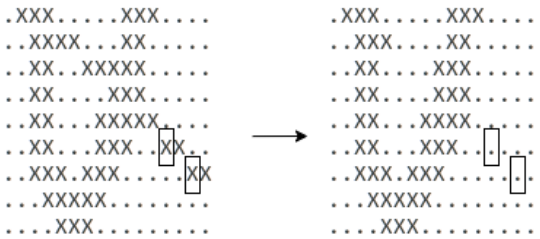


Figure 8: Process of Removing Crossed-out Noise

4.1.2 Images with overlapping digits

Several images have overlapping digits. These images could cause errors in recognition because it will be detected only as a single-digit during the segmentation process. The solution to this problem is using the number of digits that have been segmented. If the number of digits that have been segmented is less than 6, the regions that have a width greater than the average of total width will be separated into two or more digits.

4.1.3 Size normalization

Size normalization or resizing images is a process to adjust the image size into a specific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

width and height. Size normalization is performed to digits that have been obtained so that all digits will have the same number of features to be input in the classification process. The used algorithm in this resizing process is Nearest Neighbor Image Scaling.

Nearest neighbour is the simplest and fastest approach of image scaling technique. This method simply determines the "nearest" neighbouring pixel and assumes its intensity value. Nearest neighbour is the applicable algorithm to solve the SPOJ HIR problem which has a binary image as input so that there are no new pixel values other than black or white generated in the interpolation process. The digits will be resized into 16x24 pixels.

4.2 Digit Classification by Using ANN

Artificial Neural Network is a classification algorithm that can be described as a mathematical and computational model for the non-linear approximation function. The approach to solving this SPOJ Hard Image Recognition (HIR) problem is to use the weights of ANN that have been trained to determine the output of the image.

The training dataset has been made using fonts that have been downloaded through a website that provides various types of fonts [14]. Besides, the dataset was also taken from the SPOJ HIR open contest forum discussion [8]. The dataset consists of digits from 0 to 9 using different fonts and in several different sizes.

This paper will perform two network architectures. The first architecture uses only 3 layers consisting of the input layer, 1 hidden layer, and the output layer. The hidden layer has *n*-nodes which every node has a weight and single bias. The value of *n* in the hidden layer can be optimized using a Genetic Algorithm. The output layer has *k*-nodes which correspond to the class that is 10 (0-9). The architecture's illustration is shown in Figure 9.

The second architecture uses 4 layers consisting of the input layer, 2 hidden layers, and the output layer. The value of *n* in each of the hidden layers will also be optimized using Genetic Algorithm. In this paper, there are limitations of the number of the hidden layers and the value of *n* in the hidden layers considering the source code limit on the test program which later will be submitted to the online judge. The maximum number of the hidden layer is 2 and the maximum number of the value of *n* in the hidden layers is 63.

After obtaining weights and biases from the training results, weights and biases are extracted to be put on the test program to be submitted to the online judge. The test program uses the same architecture as the training program.

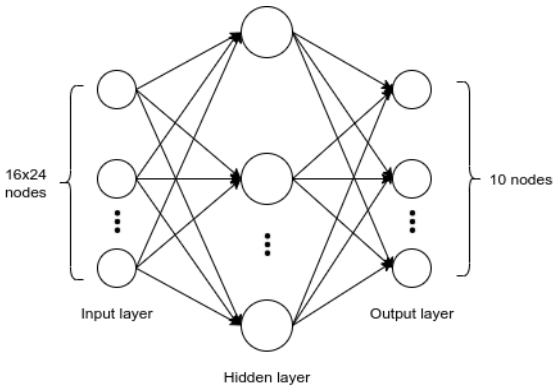


Figure 9: ANN Architecture for Digit Classification

4.2.1 Activation function

An activation function is a function for limiting the amplitude of the output of a neuron. The activation function is also referred to as a squashing function, in that it squashes (limits) the range of the output signal to some finite value [15].

Essentially, the normalized output range after being entered into the activation function will be in the range [0, 1] or [1, 1]. In this paper, the activation function that is used is the logistic activation function or also called the sigmoid activation function.

The advantage of using the sigmoid activation function is that the sigmoid function is nonlinear. In other words, when multiple neurons are having a sigmoid function as their activation function, the output is non-linear as well.

4.2.2 Optimization function

The optimization function is a function used to optimize the objective function. In the ANN, the optimization function is used to optimize the weight by minimizing the error of the prediction results to the desired output. In this paper, the optimization function used is the Adaptive Moment Estimation (Adam) optimization function.

Adam uses an adaptive learning rate for the weight and bias of each neuron in ANN. Unlike the SGD algorithm which uses the same learning rate every time, the learning rate on Adam's optimization function adapts during training.

4.3 Optimization ANN Architecture by Using GA

The value of *n* on the hidden layer of ANN architecture that has been built is a parameter that

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

can be optimized. The value of *n* can vary from 1 to infinity. In this paper, GA is used to search the solution space for the ''best'' value of *n*.

The process of GA begins with a set of individuals which is called a population, which is consisting of several chromosomes arranged by several genes which are representations of candidate solutions to a problem. In this paper, the populations used in each of the generations were 8 chromosomes. For the first architecture which is using only 1 hidden layer, every chromosome is expressed by a 6-binary string that represents the number of neurons in the hidden layer. Due to the binary coding representation, the number of neurons in each hidden layer is found in the interval [0,63]. The example of the chromosomes is shown in Table 1.

On the other hand, on the second architecture which is using 2 hidden layers, every chromosome is expressed by a 12-binary string. The first group of 6 bits correspond to the number of neurons in the first hidden layer and the second group of 6 bits correspond to the number of neurons in the second hidden layer. The example of the chromosome representation is shown in Table 2.

Table 1: Examples of Binary Chromosomes and Their Decimal Representation for 1 Hidden Layer

1st Chromosome	
$n = 34$	1 0 0 0 1 0
2nd Chromosome	
$n = 60$	1 1 1 1 0 0
...	
8th Chromosome	
$n = 19$	0 1 0 0 1 1

Table 2: Examples of Binary Chromosomes and Their Decimal Representation for 2 Hidden Layer

1st Chromosome											
1	0	0	0	1	0	1	0	0	1	0	0
$n_1 = 34$						$n_2 = 36$					
2nd Chromosome											
0	0	1	0	0	1	0	0	0	1	0	0
$n_1 = 9$						$n_2 = 4$					
...											
8th Chromosome											
0	0	1	0	0	1	0	0	0	1	0	0
$n_1 = 9$						$n_2 = 4$					

4.3.1 Objective function

The objective function is formulated by combining training error and generalization error. This is the function whose value must be minimized by the GA and that characterizes the fitness of each ANN. The objective function in GA can be calculated by Eq. (3).

$$F(i) = \alpha e^{-(L_{training(i)} + L_{generalization(i)})} \quad (3)$$

Where α is constant and e is a natural exponential function. The loss function that is used is the logistic loss or also known as a cross-entropy function. It is far more robust and more meaningful than by using the accuracy score.

4.3.2 Crossover

Basic strategy used in GA to create the best solutions/offspring is to crossover the parent genes. The two-points crossover was used to produce the offspring. The offspring size is the number of the population except for both parents. The example of this process is shown in Figure 10.

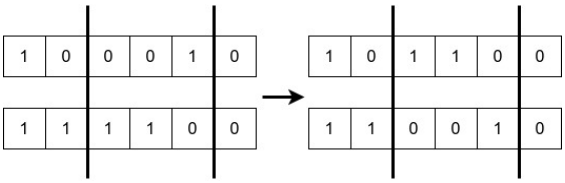


Figure 10: Crossover Process

4.3.3 Mutation

The second GA variant, mutation, is applied to the results of the crossover stored in the offspring. It is used to produce diversity in the population and usually applied with a low probability. In this paper, a bit flip mutation is applied to 1 gen of 50% chromosome in the offspring. The example of this process is shown in Figure 11. The next generation will carry the new population by appending both parents and offspring.

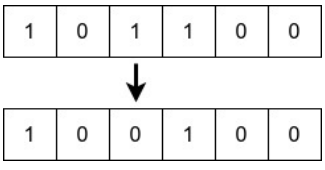


Figure 11: Mutation Process

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4.3.4 Parent selection

The selection process aims to select the best parents and recombine to create offspring for the next generation. The fitness value of each chromosome in the population will be calculated, then the 2 candidates will be selected as parents using Rank based Selection.

In rank-based selection, the population has been sorted according to the fitness values. It results in slow convergence but it prevents too quick convergence. The rank-based selection also keeps up selection pressure when the fitness variance is low [16].

5. Evaluation

In this section, the proposed GA optimization technique is used to obtain the best value of *n* on the hidden layer of ANN architecture for the SPOJ HIR problem. The populations used in each generation were 8 chromosomes. The parameters of neural network architecture and the parameters of the genetic operations are listed in Table 3 and Table 4, respectively.

The Optimization ANN Architecture using GA has been implemented using two languages, i.e., Python 3.6 and C++. The training process has been implemented in Python then its result will be extracted to be put on the test program written in C++ to be submitted to SPOJ. Two types of tests will be performed to evaluate the implemented program: the performance test and the correctness test.

Table 3: Parameters of the ANN Architecture

Parameters	Value
Maximum iterations	1000
Activation function	Sigmoid
Optimization function	Adam
Learning rate	0.01

Table 4: Parameters of the Genetic Operations

Parameters	Value
Maximum generations	300
Length of chromosomes	6, 12
Parent selection size	2
Crossover probability	75%
Mutation probability	50%
Genes mutated	1

On one hand, the performance test is done by measuring the performance of the GA to optimize the number of neurons from the ANN. The test has been performed by comparing the results using the predetermined number of neurons in the hidden layers obtained from the rule-of-thumb and the results using the number of neurons that have been optimized by GA. By using the rule of a geometric pyramid [6], for the network with 1 hidden layer with *n* neurons in the input layer and *m* neurons in the output layer, the number of neurons in the hidden layer can be calculated by Eq. (4).

$$N = \sqrt{mn} \tag{4}$$

For the network with 2 hidden layers, the number of neurons in the first hidden layer and the second hidden layer can be calculated by Eq. (5) and Eq. (6), respectively.

$$N_1 = m^{\frac{1}{3}} n^{\frac{2}{3}} \tag{5}$$

$$N_2 = m^{\frac{2}{3}} n^{\frac{1}{3}} \tag{6}$$

The number of the input layer is 384 neurons, thus in this experiment, the number of neurons for 1 hidden layer architecture is 62 and the number of neurons for 2 hidden layers architecture are 114 and 34.

On the other hand, the correctness test is performed by submitting two source codes to the problem in Sphere Online Judge: Hard Image Recognition (HIR). The first code is using optimized 1 hidden layer architecture while the second code is using optimized 2 hidden layer architecture. The source codes are, then, executed and the results from the execution are compared to the answers provided by the problem maker. The correctness test is also performed offline to evaluate training accuracy on the dataset and to evaluate the correctness of crossed-out characters.

6. Result

The two different ANN architectures were evaluated. The first architecture uses only 1 hidden layer, while the second architecture uses 2 hidden layers. For each architecture, five different runs of the GA were executed to reduce the uncertainty of performance. After each run, the ANN model with the highest objective function in each generation was stored and the execution time was recorded.

The results for all five runs of the first architecture are presented in Table 5. The best of the five runs gave 60 for the number of neurons in

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

the hidden layer. This architecture was chosen to be evaluated by SPOJ.

Table 5: Results of the Five Runs (First Architecture)

Run no	The best number of neurons	Highest objective value	Execution time (s)
1_1	53	517.440	370
1_2	33	536.509	361
1_3	60	544.056	715
1_4	48	534.969	409
1_5	60	544.056	329

The history of the best objective function value, corresponding to run number 1_5, is displayed in Figure 12.

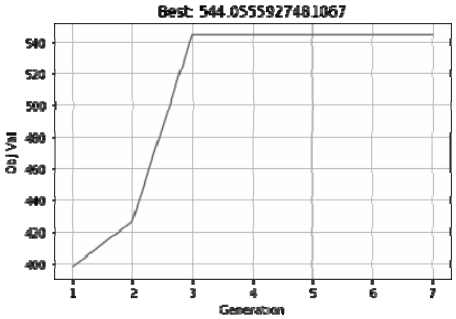


Figure 12: History Objective Value of Run Number 5 of the First Architecture

The results for all five runs of the second architecture which is using two hidden layers are presented in Table 6. The best of the five runs gave 54 and 46 for the number of neurons in the hidden layer 1 and hidden layer 2, respectively. This architecture was chosen to be evaluated by SPOJ.

Table 6: Results of the Five Runs (Second Architecture)

Run no	The best number of neurons	Highest objective value	Execution time (s)
2_1	(57, 21)	530.931	532
2_2	(33, 20)	494.043	380
2_3	(59, 51)	509.612	473
2_4	(54, 46)	564.499	555
2_5	(63, 46)	507.907	455

The history of the best objective function value, corresponding to run number 2_4, is displayed in Figure 13.

The program with the elimination of crossed-out on the pre-processing stage has successfully eliminated crossed-out noise and produces the correct output. The example of input-output of this process can be seen in Figure 14a and Figure 14b.

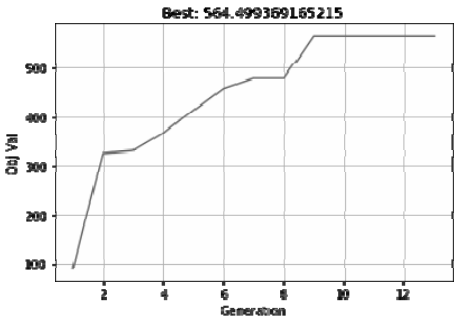
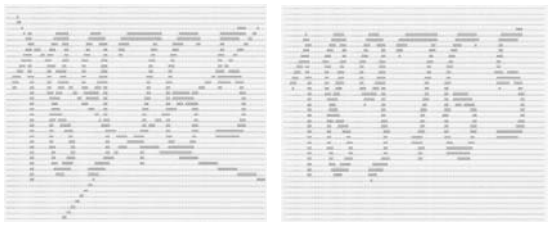


Figure 13: History Objective Value of Run Number 4 of the Second Architecture



(a) (b)

Figure 14: Process of Eliminating Crossed Out: (a) Input and (b) Output

Figure 15-16 shows the results using the predetermined number of neurons in the hidden layers obtained from the rule-of-thumb. Figure 15 shows the results using 62 for the number of neurons in 1 hidden layer architecture and Figure 16 shows the results using 114 and 34 for the numbers of neurons in 2 hidden layer architecture.

Figure 15: Verdict for submissions using 62 for the number of neurons in the 1 hidden layer architecture

Figure 16: Verdict for submissions using 114 and 34 for the numbers of neurons in the 2 hidden layer architecture

Figure 17-18 shows how the approaches using the selected ANN architectures are scored in Sphere

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Online Judge. Each source code submission has been executed at least 5 times to ensure the solution's both validity and consistency during the correctness test. Based on Figure 17, the first code which is using optimized 1 hidden layer architecture, achieves score 108. This source code has an average runtime of 0.12 seconds and average memory consumption of 5.06 MB. Figure 18 shows the second code which is using optimized 2 hidden layer architecture. This approach achieves score 100 and has an average runtime of 0.124 seconds and average memory consumption of 5.08 MB. It can be seen that the score for the number of neurons that have been optimized by GA is higher than the score for the number of neurons obtained from the rule of the geometric pyramid.

Figure 19 shows how the solution ranked among users in Sphere Online Judge. It is shown that the first implementation, using 1 hidden layer architecture, takes the best solution to the first rank on the submission.

ID	DATE	USER	RESULT	TIME	MEM	LANG
26109915	2020-06-07 16:41:08	Yolanda Hertita	108 edit ideone it	0.12	5.0M	CPP14
26109912	2020-06-07 16:40:40	Yolanda Hertita	108 edit ideone it	0.12	5.1M	CPP14
26109908	2020-06-07 16:40:20	Yolanda Hertita	108 edit ideone it	0.12	5.0M	CPP14
26109899	2020-06-07 16:37:52	Yolanda Hertita	108 edit ideone it	0.12	5.1M	CPP14
26109894	2020-06-07 16:37:42	Yolanda Hertita	108 edit ideone it	0.12	5.1M	CPP14

Figure 17: Verdict for Submissions by Using the First ANN Architecture

ID	DATE	USER	RESULT	TIME	MEM	LANG
26109418	2020-06-07 19:13:21	Yolanda Hertita	100 edit ideone it	0.13	5.2M	CPP14
26109416	2020-06-07 19:13:07	Yolanda Hertita	100 edit ideone it	0.12	5.0M	CPP14
26109414	2020-06-07 19:12:54	Yolanda Hertita	100 edit ideone it	0.12	5.1M	CPP14
26109411	2020-06-07 19:12:24	Yolanda Hertita	100 edit ideone it	0.13	5.0M	CPP14
26109407	2020-06-07 19:11:40	Yolanda Hertita	100 edit ideone it	0.12	5.1M	CPP14

Figure 18: Verdict for Submissions by Using the Second ANN Architecture

RANK	DATE	USER	RESULT	TIME	MEM	LANG
1	2020-07-09 10:57:26	Yolanda Hertita	108	0.12	5.1M	CPP14
2	2019-11-24 08:19:53	Rully Soelaiman	107	0.12	5.0M	CPP
3	2018-12-25 09:25:39	CynthiaDewi	32	0.07	17M	CPP14
4	2015-08-24 11:44:05	::Manish Kumar::	2	0.00	2.2M	C
5	2011-01-21 20:16:41	Satyarth Kumar Prasad	2	0.00	2.6M	C++ 4.3.2

Figure 19: Solution Ranking in SPOJ HIR Problem

7. Conclusion

In this paper, we presented an experiment to see if a genetic algorithm can be used to automatically generate good ANN architectures without any human intervention. The creation of the dataset and the algorithm development starting from the preprocessing stage until the digit classification using ANN can achieve a high score 108 on SPOJ by producing the correct output with runtime 0.1 seconds, approximately. The score obtained by using GA for optimizing the number of neurons of ANN architecture is higher than the score for the number of neurons obtained from the rule-of-thumb.

This research has not far yet from over since the maximum score on SPOJ HIR problem is 250. Further research is needed to provide a better solution for this problem so it can achieve higher accuracy and it can be applied to the system with limited computer hardware resources.

REFERENCES:

[1] E. Tuba, R.C. Hrosik, A. Alihodzic, R. Jovanovic and M. Tuba, "Support Vector Machine Optimized by Fireworks Algorithm for Handwritten Digit Recognition", In: *Modelling and Development of Intelligent Systems 6th International Conference*, Sibiu, Romania, pp. 187-199, 2019.

[2] M. R. Phangtriastu, J. Harefa and D.F. Tanoto, "Comparison between neural network and support vector machine in optical character recognition". In: *Procedia Computer Science of 2nd International Conf. on Computer Science and Computational Intelligence*, pp. 351-357, 2017.

[3] S. Haykin, "Neural Networks and Learning Machines", Pearson Education, Upper Saddle River, NJ, 2009.

[4] SPOJ. (2015). HIR - Hard Image Recognition, [Online]. Available: https://www.spoj.com/problems/HIR/.

[5] C. Sombattheera, K.L. Nguyen, R. Wankar, and T. Quan, Multi-disciplinary Trends in Artificial Intelligence: 6th International Workshop, Springer Publishing Company, Vietnam, 2012.

[6] T. Masters, "Practical neural network recipes in C++", Academic Press Professional, Inc., USA, 1993.

[7] PG. Benardos and GC. Vosniakos, "Optimizing feedforward artificial neural network architecture", *Engineering Applications of Artificial Intelligence*, Vol.20, pp.365-382, 2007.

[8] Discuss SPOJ. (2006). Hard Image Recognition Open Contest 2006, [Online]. Available: http:// discuss.spoj.com/t/hard-image-recognition/663

[9] T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein, "Introduction to Algorithms", 3rd edition, MIT Press, United States, 2009.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[10] Anonim. (2009). Nearest Neighbor Image Scaling, [Online]. Available: http://tech-algorithm.com/articles/nearest-neighbor-image-scaling/.

[11] A. P. Engelbrecht, "Fundamentals of computational swarm intelligence", First Edition, John Wiley & Sons, NJ, United States. 2006.

[12] B. Coppin, "Artificial Intelligence Illuminated", First Edition, Jones and Bartlett Publishers, Sudbury, United States, 2004.

[13] A. Brink, H. van der Klauw, and L. Schomaker, "Automatic removal of crossed-out handwritten text and the effect on writer verification and identification", *Proc. of Document Recognition and Retrieval SPIE International Symposium on Electronic Imaging*, San Jose, USA, 2008.

[14] Dafont Authors. (2000). Dafont, [Online]. Available: https://www.dafont.com/.

[15] S. Haykin, "Neural Networks and Learning Machines", Pearson Education, Upper Saddle River, NJ, 2009.

[16] L.S. Yadav and A. Sohal. "Comparative Study of Different Selection Techniques in Genetic Algorithm". *International Journal of Engineering Science*, Vol.6, 2017.