An Approach to Solve Sudoku Using Image Processing and with Genetic Algorithm

Amit Pokhrel[1], Irajan Dhakal[1], Sagar Bhandari[1], Sahasramshu Pokhrel[1*], Ashok GM[2], Himal Chand Thapa[3]
[1]*Himalaya College of Engineering, Chyasal, Nepal*
[2]*Head, Electronics and Computer Department, Himalaya College of Engineering*
[3]*Head, Computer Science and IT Department, Himalaya College of Engineering*
Corresponding Email*: sahaspok@gmail.com

Abstract

Sudoku is logic-based puzzle and quite a popular game. The 9 by 9 square grid Sudoku puzzle is the one that we have taken into an account. For solving sudoku puzzle, we have used image processing along with genetic algorithm. In the first phase, the raw and unsolved image is first gray scaled and then Gaussian blur is applied to reduce noise. Then it goes through a main frame extraction stage where adaptive thresholding, maximum contour detection, perspective transformation and small grid segmentation are performed respectively. The individual 81 grids are passed to the CNN model for digit recognition. Then finally, genetic algorithm is used to find the optimal solution of Sudoku problem.

Keywords
Gaussian, Genetic, Adaptive thresholding

1. INTRODUCTION

A Sudoku puzzle consists of 81 cells which are divided into nine columns, rows and regions. The task is now to place the numbers from 1 to 9 into the empty cells in such a way that in every row, column and 3×3 region each number appears only once. A Sudoku has at least 17 given numbers but normally there are 22 to 30.

The three basic rules for playing Sudoku are:-

•Each row must contain all numbers from 1 to 9.

•Each column must contain all numbers from 1 to 9.

•Each box must contain all numbers from 1 to 9.

For manipulation of images, Image processing is used which is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image. Some of the important applications of image processing in the field of science and technology include computer vision, remote sensing, feature extraction, face detection, forecasting, optical character recognition, finger-print detection, optical sorting, argument reality, microscope imaging, etc. Image processing basically includes the following three steps: Importing the image via image acquisition tools, Analyzing and manipulating the image, Output in which result can be altered image or report that is based on image analysis. In our project, image processing is mainly used for features such as Gaussian blur, adaptive thresholding, main frame detection, perspective transformation, grid segmentation, digit extraction and many more.

During digit recognition stage, Convolutional Neural Network (CNN) model is used. CNN is a type of artificial neural network used in image recognition and processing that is specifically designed to process large pixel data. Neural Networks mimic the way our nerve cells communicate with interconnected neurons and CNNs have a similar architecture. What makes them unique from other neural networks is the convolutional operation that applies filters to every part-of the previous input in order to extract patterns and features maps. In our project, a pre-trained model CNN model is used for MNIST handwriting digit recognition considering 28*28 pixels.

2. LITERATURE REVIEW

Muhammad Usman Tariq and Sara Aqab in their study "Handwriting Recognition using Artificial Intelligence Neural Network and Image Processing" [1]writes about Handwriting character recognition which refers to the computer's ability to detect and interpret intelligible. Handwriting input from Handwriting sources such as touch screens, photographs, paper documents, and other sources. Handwriting characters remain complex since different individuals have different handwriting styles. This paper aims to report the development of a Handwriting character recognition system that will be used to read students and lectures Handwriting notes. The development is based on an artificial neural network, which is a field of study in artificial intelligence. Different techniques and methods are used to develop a Handwriting character recognition system. However, few of them focus on neural networks. The use of neural networks for recognizing Handwriting characters is more efficient and robust compared with other computing techniques. The paper also outlines the methodology, design, and architecture 0 of the Handwriting character recognition system and testing and results of the system development. The aim is to demonstrate the effectiveness of neural networks for Handwriting character recognition.

Hung-Hsuan Lin and I-ChenWu in their study "An Efficient Approach To Solving The Minimum Sudoku Problem" focused on an open problem to find the minimum-clue Sudoku puzzles, also commonly called the minimum Sudoku problem [2]. Solving the problem can be done by checking 5,472,730,538 essentially different Sudoku grids independently or in parallel. In the past, the program CHECKER, written by McGuire, required about 311 thousand years on one-core CPU to check these grids completely, according to our experimental analysis. This paper is to propose a more efficient approach to solving this problem. We design a new algorithm, named disjoint minimal unavoidable set (DMUS) algorithm, to help solve the minimum Sudoku problem more efficiently. After incorporating the algorithm into the program and further tuning the program code, our experiments showed that the performance was greatly improved by a factor of 128.67. Hence, it is estimated that it only takes about 2417 years to solve the problem. Thus, it becomes feasible and optimistic to solve this problem by using a volunteer computing system, such as BOINC.

Agnes M. Herzberg and Ram Murty in their research article "Sudoku squares and chromatic polynomials" interprets the nine by nine Sudoku puzzle as a graph coloring problem [3]. Viewed this way, they can generalize the puzzle to arbitrary size. They compute the chromatic number of the resulting graph, and they estimate the number of “Sudoku squares” of any given size.

Shamsul K Khalid and Mustafa M Deris, purposed “A Systematic Redundancy Approach in Watermarking Using Sudoku" [4]. Digital watermarking is one of the most important techniques employed to protect digital contents. Copyright information embedded in the digital contents may become undetectable due to compression and additive noise attacks introduced into a watermarked image. A general strategy to overcome such attacks is to embed redundant watermarking to improve the survivability of the watermark. However, current approaches in redundant watermarking is generally not designed in a systematic way. In this paper, a systematic redundancy approach in watermarking using Sudoku is proposed. The approach is based on the property of Sudoku's permutation that provide tightly related, redundant copies of watermark pieces discretely distributed to different hiding planes. Using the classic 9X9 Sudoku puzzle, the proposed approach has been implemented and tested with Gaussian, compression, Poisson, speckle, low pass filter, high pass filter and contrast attacks using standard images. The result indicates sustained survivability of the watermark under common attacks.

Some hobbyists have developed computer programs that will solve Sudoku puzzles using a backtracking algorithm, which is a type of brute force search. Backtracking is a depth-first search (in contrast to a breadth-first search), because it will completely explore one branch to a possible solution before moving to another branch. Although it has been established that approximately 5.96 x 1126 final grids exist, a brute force algorithm can be a practical method to solve Sudoku puzzles A brute force algorithm visits the empty cells in some order, filling in digits sequentially, or backtracking when the number is found to be not valid.

OpenCV (Open-Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code. The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish

markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library is used extensively in companies, research groups and by governmental bodies. Along with well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library, there are many startups such as Applied Minds, VideoSurf, and Zeitera, that make extensive use of OpenCV. OpenCV's deployed uses equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.

In image processing, a Gaussian blur (also known as Gaussian smoothing) is the result of blurring an image by a Gaussian function (named after mathematician and scientist Carl Friedrich Gauss) [5]. It is a widely used effect in graphics software, typically to reduce image noise and reduce detail. The visual effect of this blurring technique is a smooth blur resembling that of viewing the image through a translucent screen, distinctly different from the bokeh effect produced by an out-of-focus lens or the shadow of an object under usual illumination. Gaussian smoothing is also used as a pre-processing stage in computer vision algorithms in order to enhance image structures at different scales—see scale space representation and scale space implementation. A Gaussian blur effect is typically generated by convolving an image with a FIR kernel of Gaussian values.

In recent times, with the increase of Artificial Neural Network (ANN), deep learning has brought a dramatic twist in the field of machine learning by making it more artificially intelligent. Deep learning is remarkably used in vast ranges of fields because of its diverse range of applications such as surveillance, health, medicine, sports, robotics, drones, etc. In deep learning, Convolutional Neural Network (CNN) is at the center of spectacular advances that mixes Artificial Neural Network (ANN) and up to date deep learning strategies. It has been used broadly in pattern recognition, sentence classification, speech recognition, face recognition, text categorization, document analysis, scene, and handwritten digit recognition. The goal of this paper is to observe the variation of accuracies of CNN to classify handwritten digits using various numbers of hidden layers and epochs and to make the comparison between the accuracies. For this performance evaluation of CNN, we performed our experiment using Modified National Institute of Standards and Technology (MNIST) dataset. Further, the network is trained using stochastic gradient descent and the back propagation algorithm [6].

3. METHODOLOGY

3.1 Image Processing

Grey Scaling

The three basic levels of grey level transformation involve linear, logarithmic and power-law transformation. Linear transformation in grey scaling includes simple identity and negative

transformation. The second linear transformation is negative transformation, which is an invert of identity transformation. In negative transformation, each value of the input image is subtracted from the L-1 and mapped onto the output image. Gray Scaling has been deployed to detect the contour over the project. Better Computer vision shall be attained using the grey scaling. Gray scaling is done using the formula as,

RGB[A] to Gray: Y ⟵ 0.299 * R + 0.587 *G + 0.114 * B …..(Equation 1)

Where, Y is the pixel value for given pixel position

R is intensity of Red color at given pixel position

G is intensity of Green color at given pixel position

B is intensity of Blue color at given pixel position

Gaussian Blur

When we blur an image, we make the color transition from one side of an edge in the image to another smooth rather than sudden. The effect is to average out rapid changes in pixel intensity. The blur, or smoothing, of an image removes "outlier" pixels that may be noise in the image. Blurring is an example of applying a low-pass filter to an image. In computer vision, the term "low-pass filter" applies to removing noise from an image while leaving the majority of the image intact. A blur is a very common operation we need to perform before other tasks such as edge detection. There are several different blurring functions in the skimage. filters module, so we will focus on just one here, the Gaussian blur. In short, we need to gaussian blur the image to reduce noise in the thresholding algorithm. The Gaussian blur algorithm implemented over the project utilizes the mathematical model. The formula of a Gaussian function in two dimensions is

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \text{.....(Equation 2)}$$

Where σ is the standard deviation of Gaussian distribution(usually taken as 1)

x is the distance from the origin in the horizontal axis

y is the distance from the origin in the vertical axis

The aforementioned formula gives 5*5 low pass Gaussian filter as,

$$\frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

Figure 1: 5×5 low pass Gaussian filter

Adapting Thresholding

Adaptive thresholding calculates the threshold value for smaller regions of 11*11 px and therefore, there will be different threshold values for different regions. The mean of these different region is calculated and threshold value is calculated respectively. Thresholding is a type of image segmentation, where we change the pixels of an image to make the image easier to analyze. In thresholding, we convert an image from color or grayscale into a binary image, i.e., one that is simply black and white. This would make our image preprocessing task simpler to process.

$$dst(x,y) = \begin{cases} maxValueis\ \ src(x,y) > T(x,y) \\ 0 \qquad\qquad otherwise; \end{cases}$$

……(Equation 3)

Where,

dst(x,y) = destination image

src(x,y) = source image

T(x,y) = Threshold value calculated individually for each (x,y) pixel as mean(blockSize * blockSize of neighbour - 2)

3.2 Main Frame Extraction

Contour Detection

Contour detection simply identifies the bounded areas over the image in user input. In simpler words, there could exist multiple bounded areas available in the image. This might include the sudoku puzzle header and Puzzle frame itself. Contour detection simply identifies these bounded areas from the image. First one is source image, second is contour retrieval mode, third is contour approximation method. And it outputs a modified image, the contours and hierarchy. contours is a Python list of all the contours in the image. Each individual contour is a NumPy array of (x,y) coordinates of boundary points of the object.

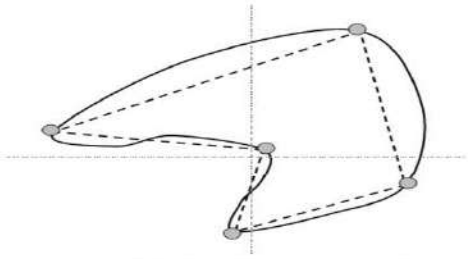


Figure 2: Polygon approximation of a contour

Maximum Puzzle Frame Determination

The non-essential aspects include the text that comprises heading and other non-digit characters that are not useful. Maximum Puzzle Frame thus identifies the biggest contour. After contour detection, different pre-defined polygons are drawn inside the contour using polygon approximation method. Then areas of different polygon are calculated respectively. The largest area of the polygon is selected as a frame. User submitted images might comprise non-essential aspects of sudoku as well. Also, the contour sometime may require the perspective transformation if the image captured has been skewed essential transformations are carried out.

Perspective Transform

The image orientation could be skewed or inclined at any angle i.e. the captured image may not be horizontally captured. To address the situation the perspective transform has been deployed so that the image is aligned appropriately for easier processing of the image. Perspective transform hereby shall result into the formation of 900*900 pixel image.

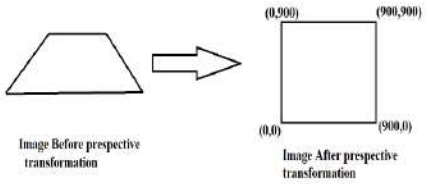


Figure 3: Perspective Transformation of an Image

3.3 Small Grid Segmentation

In order for any digital computer processing to be carried out on an image, it must first be stored within the computer in a suitable form that can be

manipulated by a computer program. Most commonly, the image is divided up into a rectangular grid of pixels, so that each pixel is itself a small rectangle. Being specific to the project, out of 900*900 pixel image extracted previously now the grid with 100*100 pixels are identified and 4px*4px of images are clipped away. Clipping has been done so that the lines of the grid are removed. Now these clipped square boxes are resized over 28*28 pixels, this could be visualized as the detection of Sudoku digits with the least number of edges in the image.

3.4 Convolution Neural Network

The project hereby deploys the CNN in the digit identification. Deploying the MNIST dataset, the model has been trained thoroughly, the train test data has been splitted for the precise result of the model. Now once trained, model has been fed with the data i.e. images that needs to be predicted from the model in the form of text. In other words the digit from image needs to be identified and returned back as text. Once identifying the digits, the text file has been prepared accordingly to the identified digits and is exported for further processing. These exported data comprise an array, where empty image or image with no digits are identified as 0 and rest are identified according to the actual digits. The array comprises of 9*9 dimensional data. Now these data are exported using JSON file format. From the input, the first network starts with a convolution whose filters_size is equal to 5*5 is implemented. It has followed by another convolution with the same filters_size. These two convolutions are followeby a Max_pooling with pool_size equal to 2*2 and then a drop_out with 0.25 probability to switch neurons off.

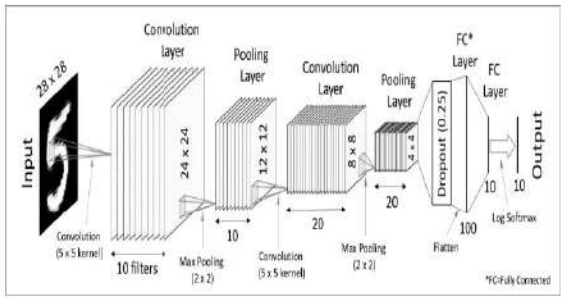


Figure 4: Convolution neural network model

3.3 Genetic Algorithm

Initialize population

At first, we determine all possible value for the cells that do not contain given values in the starting point. Then initialize population by filling that particular empty cell through determined possible value. Here, we have 1,000 initial population at the start and later on the population will be change through mutation and crossover operation but the size of the population remains same as 1000. Increased or decreased on the basis of fitness value convergence.

Fitness function

As always, coming up with the proper fitness function for the genetic algorithm is the greatest challenge. The way we determined the fitness function was using three tables(Row table Column table Square table) .The fitness algorithm first goes through each column of the Sudoku and the unique number contained in the cell is added to the ColumnMap. After an entire column is traversed, the algorithm looks at the count of the ColumnMap . If every key in the column is unique, then the ColumnMap contains a total of 9 values. If the column contains a few duplicates,

the total count in the ColumnMap will always be less than 9. The count can then be used as a representative fitness of uniqueness for a particular column. By adding this number for all columns, we can come up with a fitness for uniqueness of columns. We can then perform the same exact technique for determining row uniqueness. The RowMap is populated with keys from each cell in a row of the Sudoku grid. If the row is entirely unique, then the RowMap will contain 9 entries. The same technique is also used for each 3x3 cell grouping (shown in figure 1). The SquareMap is populated with cell values for this square table.

Fitness function for each column/row/square:

$$\text{Fitness value} = \frac{1.0}{((10 - Uniqueness)/9.0)}$$

.....(Equation 4)

i.e. Uniqueness is No. of unique entity in each chromosome

As, fitness value for each column is calculated, then they are summed up and store in fitnessCloumns. Similarly, total fitness value of row and square are summed up and store them in fitnessRows and fitnessSquares respectively. Then fitness value for the population is calculated as,

CurrentFitness = fitnessColumns * fitnessRows * fitnessSquares .....(Equation 5)

Selection

After the calculation of fitness value for each population, population are arrange on descending order based on their fitness value. Then we use the selection rate of 50%, so top 500 population with highest fitness value are selected and remaining are removed.

Crossover

Through the selected 500 population we performed the crossover operation on them in order to generate new solution. At first two population are selected on the hierarchical order then single-point crossover technique is applied to generate new child. This process is repeated until desire number of child are produced i.e. 500 for our project.

Mutation

In certain new solution is formed, then some of their genes can be subjected to a mutation based on mutation rate i.e. 0.5.At first mutation value is calculated for each genes as,

Mutation value is calculated as,

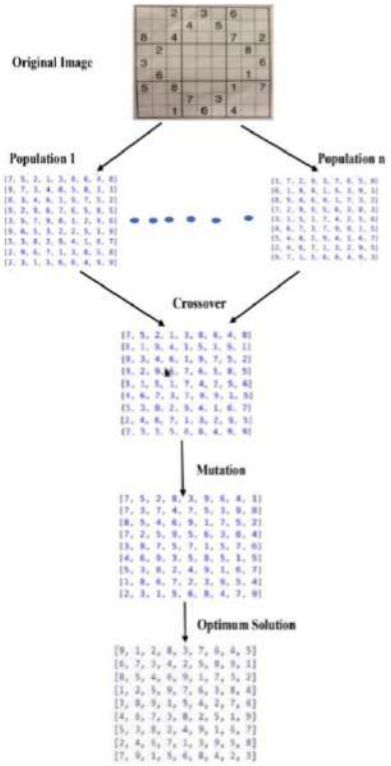
$$Mutuation\ Value = \frac{(i + j)}{18}$$

.....(Equation 6)

Where, i =Gene row position

j =Gene column position

Then if calculated mutation value is greater than mutation rate, that gene is flipped by the possible value so that our algorithm can come to a better solution efficiently.



4. RESULT

The final result and output of the project were analyzed and compared with the expected output after the successful completion of our project. When the aforementioned fitness function along with 50% mutation rate and 50% selection rate were used then at about 43 generations, the desired output of sudoku were observed. We tried to give various types of file format but the system threw an error expect for a desired image file. Puzzle was detected and solved with the help of provided sudoku image file. And in the absence of puzzle, frame can't be extracted thereby threw an error. Continuing this, pixel size of 28*28 was sent to the model in order to recognize the digit and so it did. The model couldn't able to predict and recognize the number when there was a hazy and a vague image. Grids were assigned to zero in the case of empty data. At the sudoku solving stage, a unique solution is provided when there were at least 24 predefined numbers. If the values in the same row or in the same column or in the same sub grid are same thereby violating the constraints of sudoku game then error were broadcasted with an inappropriate input. Finally, the result which won among the fitness value in the different population of genetic algorithm, came out to be the solution of the given respective sudoku problem. The graph of prediction accuracy of neural network up to 10 epoch is shown in figure 5.

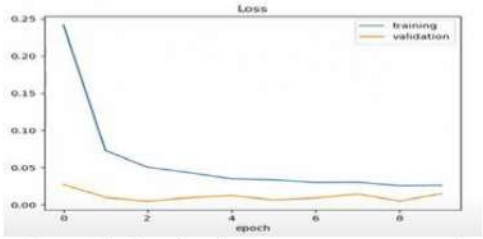


Figure 5: Graph of accuracy versus epoch

The graph of loss vs epoch is shown in figure 6.

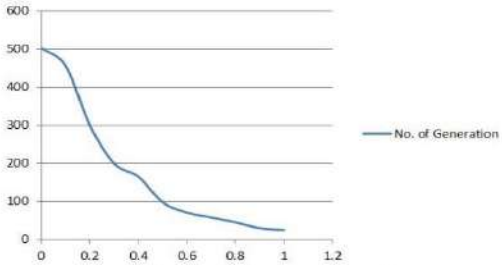


Figure 6: Loss vs Epoch

Analyzing above two graphs, we trained our model for 10 Epoch.

The graph for selection rate and Number of generation for optimum solution is shown in figure 7:

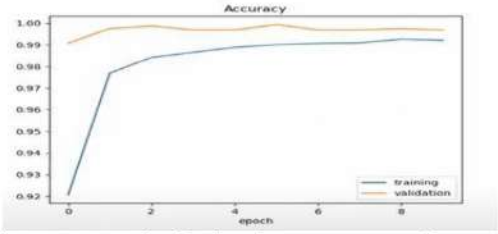


Figure 7: Graph of selection rate vs no. of generations

Here in figure 7

Y axis= No. of Generation

X axis= Selection Rate

The graph for selection rate and time complexity is shown below:

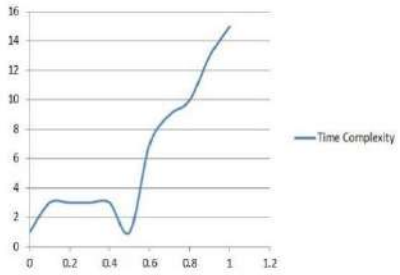


Figure 7: Selection rate Vs Time Complexity

Here in figure 8

Y axis= Time

X axis= Selection Rate

Analyzing above two graphs,0.5 selection comes out to be appropriate one since we got our output with minimum generation of population and with lesser time complexity.

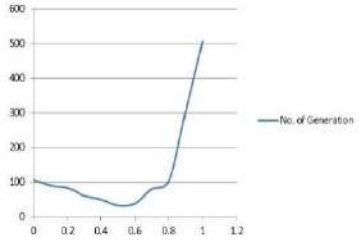


Figure 8: Mutation rate Vs Number of generation

Here in Figure 8,

Y axis= No. of Generation

X axis= Mutation Rate

Observing above graph, approximate mutation rate came to be 0.5 as the best rate.

of their role in eliminating the problems in the building permit approval system.

5. CONCLUSION

The final result of our project has been displayed on the browser. User are capable of uploading the Sudoku image that needs to be solved through the front end, once the image gets uploaded over the system several processing steps over the system are displayed as well. This would allow user to have better understanding over the system. Finally, the compiled output image is displayed back to the user again. Thus, using the portal one can solve the Sudoku problem easily with the least time being spent.

REFERENCES

[1] S. K. A. Khalid, M. M. Deris, and K. M. Mohamad, "A systematic redundancy approach in watermarking using sudoku," 2014. doi: 10.1109/ICITCS.2014.7021723.

[2] OpenCV Team, "OpenCV," 2022. https://opencv.org/about/

[3] R. K. Mohanta and B. Sethi, "A Review of Genetic Algorithm application for Image Segmentation," *International Journal of Computer Technology & Applications*, vol. 3, no. 2, 2007.

[4] J. Matthews and S. Baker, "Active appearance models revisited," *Int J Comput Vis*, vol. 60, no. 2, 2004, doi: 10.1023/B:VISI.0000029666.37597.d3.

[5] F. Siddique, S. Sakib, and M. A. B. Siddique, "Recognition of handwritten digit using convolutional neural network in python with tensorflow and comparison of performance for various hidden layers," 2019. doi: 10.1109/ICAEE48663.2019.8975496.

[6] A. E. Eiben, "Introduction to Evolutionary Computing," *Assembly Automation*, vol. 24, no. 3, 2004, doi: 10.1108/aa.2004.24.3.324.1.