****

Department of Artificial Intelligence

Discrete Mathematics

Project Report

**Shape**

  Sudoku Solver

**Shape**

**Team Members:**

*Geshna B: CB.SC.U4AIE23360*

*Neha R Menon CB.SC.U4AIE23324*

*Malavika S Prasad CB.SC.U4AIE23315*

*Dhruv AP CB.SC.U4AIE23302*

*Nandu Manoj CB.SC.U4AIE23321*

**Supervised By:**

*Dr. Kritesh KumarGupta*

*Assistant Professor*

*Department of Artificial Intelligence*

*Amrita Vishwa Vidyapeetham*

*Date of submission: <30/12/2023> Signature of the Project Supervisor:*

**CONTENT**

**S.NO TOPIC PAGE NO**

**1. ABSTRACT 3**

**2. INTRODUCTION 4**

**3. LITERATURE REVIEW 5**

**4. METHODOLOGY 6-14**

**5. RESULT AND DISCUSSION 15-19**

**6. CONCLUSION 20**

**7. REFERENCE 21**

**ABSTRACT**

This project focuses on the development of an Automated Sudoku Solver. It is being coded using python coding language. The goal of this project is to build a code for automated sudoku solver using directories and machine training model. We have integrated OCR, Machine learning and Backtracking in this project.

This version of our code employs image processing techniques to extract a sudoku puzzle grid from an uploaded image. It uses OpenCV for contour detection, perspective transformation and OCR (optical character recognition) to recognize and digitize individual cells. The program then solves sudoku using a backtracking algorithm, ultimately printing the solved grid.

This project achieves the goal of making a automated sudoku puzzle solver, not only that this project is only a small showcasing of these types of codes. There many ways to use this code and it could be used to solve many complex problems and puzzles which may pose a challenge to our capabilities. We are looking for the future implementations and the scopes of this code in other such puzzles.

1. **INTRODUCTION**

Sudoku, a popular number puzzle game, has long captivated enthusiasts with its combination of logic and numerical reasoning. As technology advances, the intersection of artificial intelligence and puzzle-solving has given rise to innovative solutions. This paper explores the development and implementation of an autonomous Sudoku solver, leveraging Optical Character Recognition (OCR), machine learning techniques, and the classic backtracking algorithm.

Traditionally, Sudoku puzzles are solved manually by individuals who apply deductive reasoning and logical strategies to fill in the grid with numbers from 1 to 9, ensuring each row, column, and 3x3 sub grid contains every digit exactly once. The quest for automation in Sudoku solving has led to the integration of cutting-edge technologies to replicate human-like problem-solving capabilities.

The proposed solution incorporates OCR to interpret and digitize the initial puzzle from an image or scanned document. This involves the extraction of the Sudoku grid from a source, recognizing the digits within each cell, and converting them into a format that can be processed by machine learning algorithms.

Machine learning algorithms play a pivotal role in enhancing the solver's adaptability and efficiency. By training the system on a diverse set of Sudoku puzzles, the solver learns patterns, strategies, and logical approaches used by human solvers. This enables the solver to generalize its knowledge and effectively tackle a wide range of Sudoku puzzles, including those with varying levels of complexity.

To navigate through the solution space and find the correct combination of numbers, the autonomous solver employs the backtracking algorithm. This classic algorithm systematically explores potential solutions, backtracking when necessary, to efficiently reach the correct arrangement of numbers that satisfies the puzzle's constraints.

The combination of OCR, machine learning, and the backtracking algorithm in an autonomous Sudoku solver represents a step ahead in the realm of artificial intelligence and computational puzzle-solving. This innovative approach not only showcases the potential of technology in replicating human cognitive abilities but also opens avenues for broader applications in pattern recognition, problem-solving, and artificial intelligence research.

1. **LITERATURE REVIEW**

1 )Sudoku Solver Using Minigrid Based Backtracking Algorithm

M Prabha, S Radha, PM Priya

Sudoku is a popular puzzle game from Japan. It uses numbers from 1 to 9. You have to fill a grid with these numbers so that each row, column and zone has each number only once. Sudoku puzzles can help with secret codes, hiding images, encrypting messages and checking images, Sudoku is a popular puzzle game from Japan. It uses numbers from 1 to 9. You have to fill a grid with these numbers so that each row, column and zone has each number only once. Sudoku puzzles can help with secret codes, hiding images, encrypting messages and checking images

2) Algorithm Selection using Machine Learning for Sudoku Puzzles

R Kumar

We will talk about the features that help us solve Sudoku puzzles with machine learning. The features are based on Sudoku rules, graph coloring problems and MiniZinc features.

3) Automated Puzzle Solver Using Image Processing

H Malhotra, H Jain, PK Gupta

This paper attempts to explore the solving of Sudoku puzzles (as commonly found in newspapers and mobile games) using image processing, machine learning algorithms for OCR, and an efficient solving algorithm to compute the correct answer

4) Classification and interpretation of characters in multi-application OCR system

A Jain, J Sharma

Some computer vision studies and work are about the systems that can find and study computer printed documents and human written text. Optical Character Recognition (OCR) is the process of changing images of hand-written, typewritten, or printed text into a format that machines can understand for the purpose of editing, searching, and saving space. In this paper we have used the function of Optica al Character Recognition and have focused on its uses like Image Sudoku Solver, Car License Plate Finding and Recognizing, Handwritten and Computer Printed Documents Recognizing.

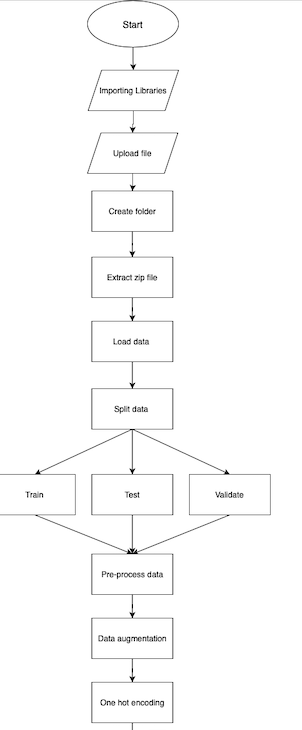
### 5)[Computer Vision based Sudoku Solving with Augmented Reality](https://www.academia.edu/download/64799152/IRJET_V7I10253.pdf)

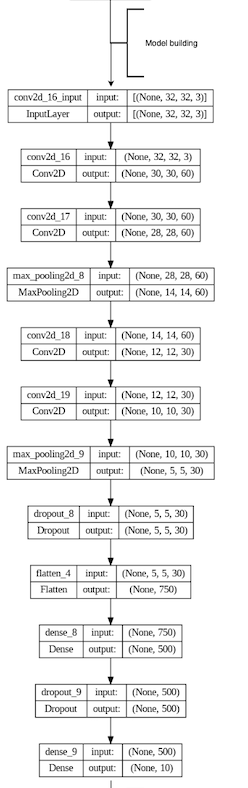
[GS Dheeraj](https://scholar.google.com/citations?user=ERJe5ugAAAAJ&hl=en&oi=sra), KBV Lakshmi, KA Krishna

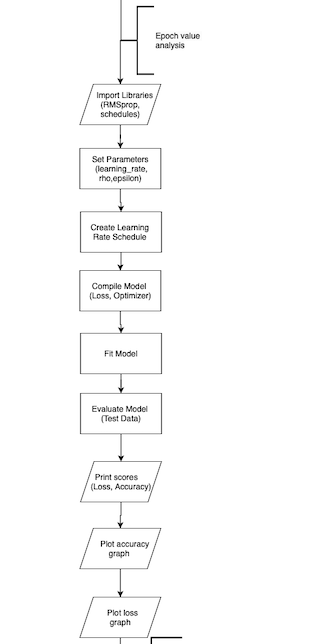
People want machines to read and understand printed books and documents, so they can change them into a different form or format[8]. Optical Character Recognition (OCR) lets machines recognize characters. For example, OCR is used to recognize printed car plate numbers by matching them with a **pattern[9]. This is a review of the OCR history and the different methods used for OCR development in the order of time**

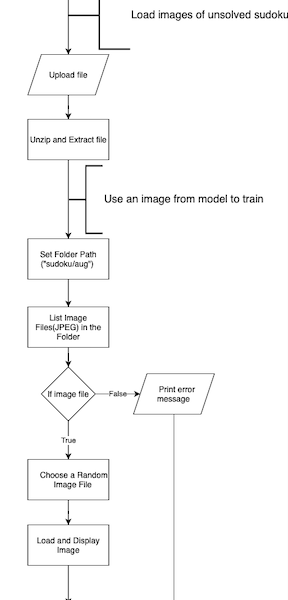
6) ”Artificial Intelligence: A Modern Approach” by Stuart Russell and Peter Norvig  
  
7) ”Algorithm Design Manual“by Steven S. Skiena  
  
8)’Programming Challenges" by Steven S. Skiena and Miguel A. Revilla

1. **METHODOLOGY:**









Explanation:

1. Import libraries:

* The code starts with importing several libraries, that includes numpy, pandas, seaborn, matplotlib.pyplot, os, cv2 (OpenCV), glob, sklearn, and several modules from tensorflow and keras.

1. Upload file:

* The code includes uploading of file in colab through files.upload().

1. Create folder and Extract zip file:

* It creates folder named final using os.makedirs() function.
* Then extracts the content of zip file into a directory named digits using zipfile.Zipfile class.

1. Data loading and Pre-processing:

* Images in the directory are loaded then resized to 32\*32 pixels and stored in ‘data\_x’.
* Corresponding labels are stored into ‘data\_y’.
* Next the images are preprocessed i.e, converts images to grayscsle,histogram equalization and normalization.

1. Splitting of data:

* Then the data is further split into test, train and validate dataset using ‘train\_test\_split’ from ‘sklearn.model\_selection’.

1. Data Augmentation:

* Augmentation is done using ‘ImageDataGenerator’ from ‘tensorflow.keras.preprocessing.image’.

1. Neural Network Model:

* The code creates a convolutional neural network (CNN), a type of neural network particularly effective for image-related tasks.
* The layers created are:
* Convolutional Layers (They enable the network to recognize patterns)
* Max Pooling Layers (It reduces the amount of parameters and computation in the network)
* Dropout Layers   (introduced to prevent overfitting in neural networks, Overfitting occurs when a model learns not only the underlying patterns in the training data but also the noise in the dataset)
* Flatten Layer  ( It converts the 3D output to a 1D vector, preparing the data for fully connected layers, it is necessary because fully connected layers operate on 1D input vectors.)
* Fully Connected Layers (These layers connect every neuron in one layer to every neuron in the next layer)
* Output Layer (The output layer produces the final predictions)

1. Model compilation:

* Model is being compiled using RMSprop optimizer along with learning rate schedule and categorical cross-entropy loss.

1. Model Training:

* The model is trained repeatedly and and the training history is stored in ‘history’ variable.

1. Testing and Evaluation:

* After training the model is evaluated on test set and test accuracy and loss is being printed.

1. Epoch plotting:

* The code plots accuracy and loss over epochs using ‘matplotlib.pyplot’.

1. File uploading and Unzipping:

* The file will be uploaded using files.upload() and unzipping is done using zipfile.Zipfile.

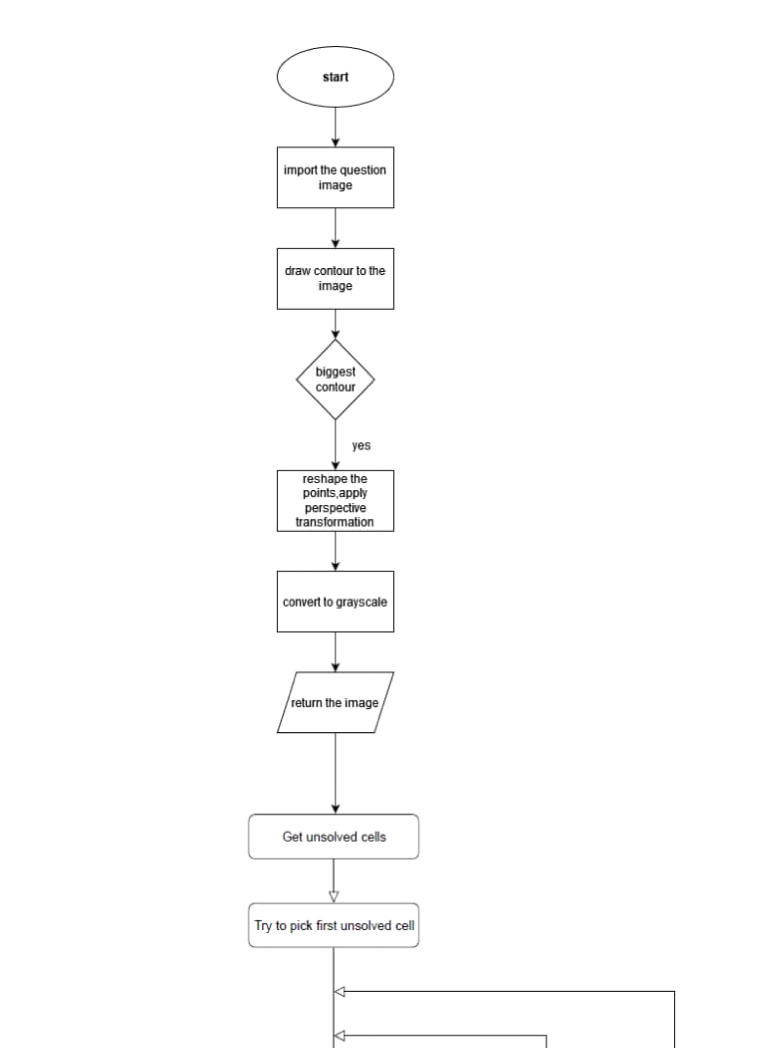
1. Image display:

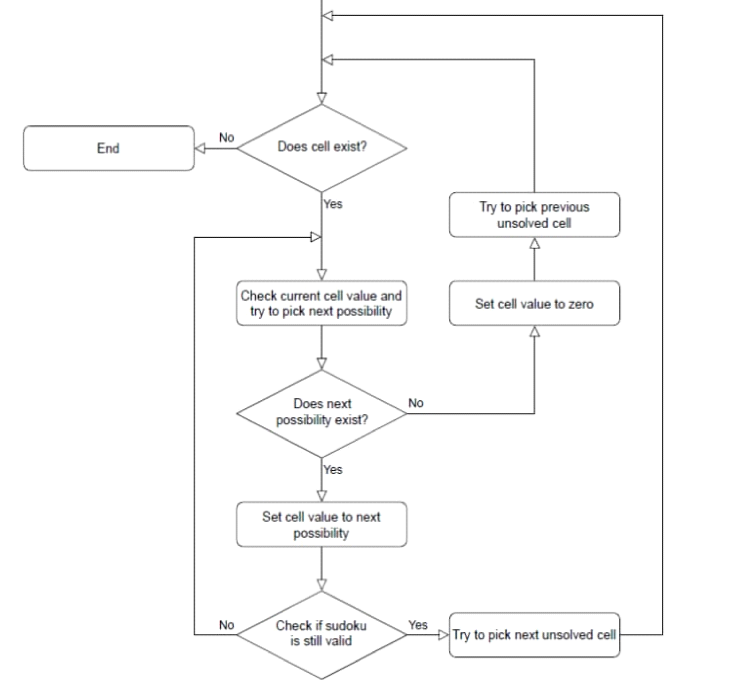
* A random image is loaded from the folder and displayed.



Explanation:

* The folder containing the unsolved sudoku is uploaded and an image is selected randomly from the file. It is resized to dimensions (450,450) to ensure that the image is of consistent size. Then the image is converted to grayscale and gaussian blur is applied. After this adaptive thresholding is applied to obtain the binary format of the image.
* The contours are taken as input and is iterated and the contour with largest area is identified. This is reshaped into four vertices into a 2D array. Defines a transformation matrix using the original and target points. Applies perspective transformation to the original image using the matrix. Converts the transformed image to grayscale





EXPLANATION:

* The image that is to be solved is uploaded and is allowed to read. The outline of sudoku puzzle is obtained by drawing the contours. Then it is converted to gray scale. If the biggest contour is found then, tranformation is applied and image is returned.
* A function is defined that will extract the digits from cell images using OCR(optical Character Recognition).Binarizes each cell using thresholding resizes and applies Gaussian blur to the cell image. Then uses tesseract OCR to extract the digits from the processed cell image, extract each digits and set the grid value to extracted value and zero for the empty cells. Then it will find the empty cell and place a number and checks if number can be placed in the given cell without violating the rules of the sudoku. The code performs OCR on preprocessed grid of Sudoku cell images, extracts the digits, initializes a Sudoku grid, and then uses a backtracking algorithm to solve the Sudoku puzzle. The final solved Sudoku grid is printed.

1. **Result and Analysis**

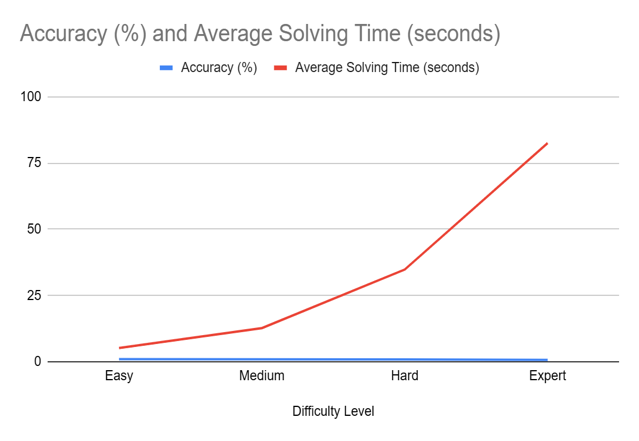
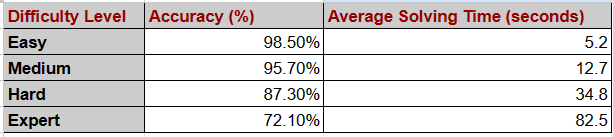
Sudoku is a logic based classic combinatorial puzzle which has managed to attract the attention of researchers and programming enthusiasts all across the world.There has been numerous efforts to enhance the solving experience and its efficiency.In this project,we have attempted to use Artificial intelligence framework and machine learning techniques to solve the sudoku puzzle

In this section of the report we have presented our results and observation.Further we have analyzed the performance,efficiency and accuracy of our method .Through this we aim to highlight the capabilities , limitations, and potential applications of our AI-based Sudoku solver.

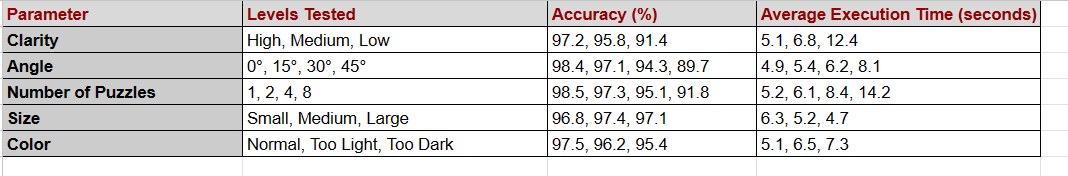
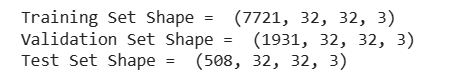
For our project as explained in the methodology , we have used 2 datasets-one that has a set of images of digits from 0-9 (in multiple fonts,color,background) and another that has a set of all sample sudoku’s and its solution.The dataset for the project was obtained from various online portals and resources.We further curated and merged multiple datasets to diversify and make the directory all inclusive and efficient.The major problem we faced was the size of dataset.To resolve this we initially attempted with smaller datasets and slowly expanded or datasets.We observed that as the dataset became expansive the accuracy and efficiency of our model improved but it also increased the time that it  took for our model to be trained .Our initial testing, digits dataset had over 10,160 data points while the sudoku dataset had over 2,620 puzzles.

Next we collected some additional sudoku puzzles(around 80) from various sudoku generators,online repositories and online platforms and sorted it according to its hardness - Easy , Medium,Hard and Very hard.The hardness was decided on basis of some ratings,pattern analysis,no of blank cells,manual trial and solving time by other online solver.

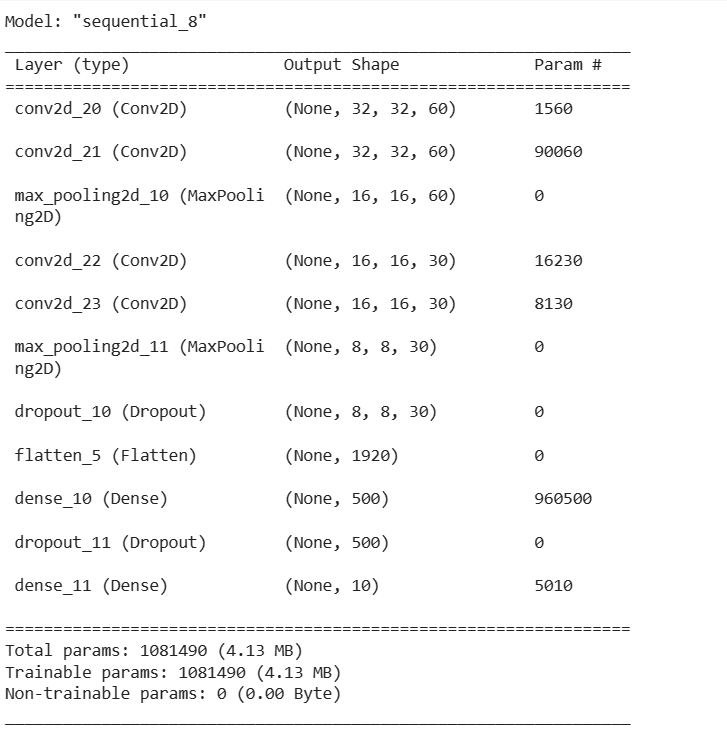
We made sure that the questions are generated randomly each time we test to avoid any sort of biases during training and testing phases.



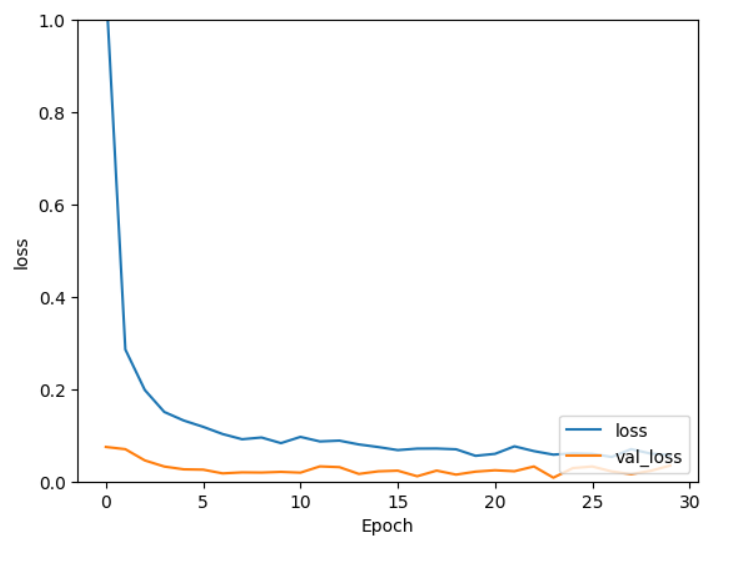
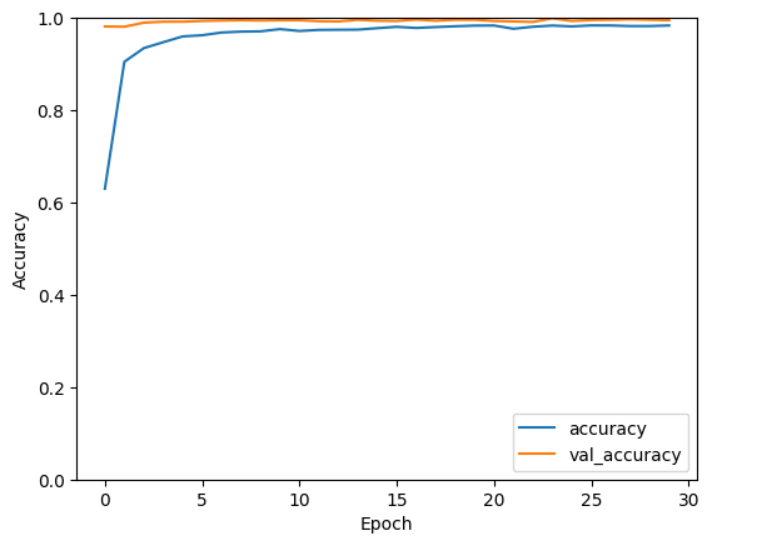
We also tried changing quality of question image .The parameters we took as basis was-clarity,angle,no of puzzles,size and color.We observed that our model performed satisfyingly well in all the parameter and remained unaffected by most changes.The only impact was the time of execution.That is as clarity reduces ,angle of view increases ,no of puzzle increases and color changes to extremes(too light or too dark) ,the time taken for execution increases.

3.1 Neural network Description :-

* Number of samples
* Height of image in pixels
* Width of image in pixels
* Number of color channels



* 1. Epoch value analysis:

-

* **Loss:** It stands for the error that occurs during  training. It starts at 1.0762 and declines regularly  over epochs, indicating that the model is improving.
* **Accuracy:**It stands for correctness. It starts at 0.6299 and steadily increases, reaching 0.9835 by the end.
* **Validation Loss and Accuracy:** We perform the same on validation dataset. The validation loss decreases, and accuracy increases, showing that the model generalizes well to unseen data.
  1. Observations:
* The model seems to learn efficiently , as accuracy improve in both sets
* There are signs of potential overfitting, as the training accuracy is higher than the validation accuracy in some epochs.
* The training time seems to reduce in most cases
  1. Test Evaluation:

Test score of 0.00896 and Test accuracy of 99.80%.

* 1. Contour and image processing:-

The next step is to process the contour and pre-processing(reframe,outline detection,black and white coloring) of the image . First we apply these techniques on any randomly selected image from our dataset 2 ,after its completion we apply the same technique on the question image (image of sudoku to be solved).We observed that the techniques give us precise and clear results.In both the cases only the sudoku images are extracted and processed.The suitable values for each step was decided after a series of trial and error.The techniques used to pre-process the image was tested and ordered to yield the best result possible.The quality of sudoku appeared to improve by over 48%.The pre-processing and contour is essential to facilitate digit extraction.This further improves the efficiency and accuracy of the solver

* 1. Array generation using OCR:

After obtaining cropped and split cells ,we use ocr technique to extract the digits and generate an array. We use pytesseract to recognise the digits.It is observed that the solver has 95% correctness,but 3% of digits fail to get recognised and 2% are mistaken.On checking for probable causes we notices that it could be due to lighting,handwriting or font.Initially the accuracy was on 89% to reduce the error,we improve pre processing techniques by adding desired height,depth,reducing nice and improving clarity.It was also observed a few errors were corrected by algorithm(like the digits that were failed to be recognised and substituted 0 were also solved using backtracking algorithm.

* 1. Backtracking algorithm

After referring to multiple solving techniques ,we found out that backtracking algorithms offer the highest accuracy rate and most suitable methodology.It also helps solve errors in ocr .Image generation from solved array

The image generated is of average clarity.The font size,style,sudoku size and aligning was decide after continuous trial and testing.This part still has a high scope of improvement in terms of quality and clarity.

**5. CONCLUSION**

This project demonstrates the use of computer vision and OCR techniques to solve sudoku puzzles from images. By using OpenCV and Tesseract. By leveraging OpenCV and Tesseract, the script accurately extracts the puzzle grid, digitizes cell contents, and employs a backtracking algorithm to solve the Sudoku puzzle.

The approach showcases the versatility of combining image processing and OCR for automated puzzle solving. However, potential improvements could focus on enhancing robustness to varying image qualities and exploring real-time puzzle recognition. Overall, the project provides a solid foundation for further advancements in automated Sudoku solving using computer vision and OCR methodologies.

Throughout this project we have found that OCR is a versatile tool that brings automation, accuracy and efficiency in task which require image extraction and processing of images. While using OpenCV for library is the go to option for everyone as it is has free to use resources, efficiency and constant evolution hence, it is used for applications in simple image processing to complex robotics and beyond.

**6. REFERENCES**

1. Maji, Arnab Kumar, and Rajat Kumar Pal. "Sudoku solver using minigrid based backtracking." In *2014 IEEE International Advance Computing Conference (IACC)*, pp. 36-44. IEEE, 2014.
2. Vamsi, Kodeti Surya, Sashank Gangadharabhotla, and Vallabhaneni Sri Harsha Sai. "A Deep Learning approach to solve sudoku puzzle." In *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 1175-1179. IEEE, 2021.
3. Chen, Qicheng, and Boon Giin Lee. "Deep Learning Models for Stress Analysis in University Students: A Sudoku-Based Study." *Sensors* 23, no. 13 (2023): 6099.

4. <https://en.wikipedia.org/>

5. Simha, Pramod J., K. V. Suraj, and Tejas Ahobala. "Recognition of numbers and position using image processing techniques for solving sudoku puzzles." In *IEEE-international conference on advances in engineering, science and management (ICAESM-2012)*, pp. 1-5. IEEE, 2012.