The Sudoku Project

WEEK 9: 18/09/2021 to 25/09/2021

Ishika De and Yashvi Donga

September 2021

Agenda

- Brief Overview
- Current Status
- Toolchain
- Learnings

Brief Overview

The goal of this project is to investigate a variety of algorithms (backtracking, brute force, stochastic search and Crook's algorithm) that are capable of solving sudoku puzzles, of ranging difficulties, in order to learn more about sudoku solving techniques.

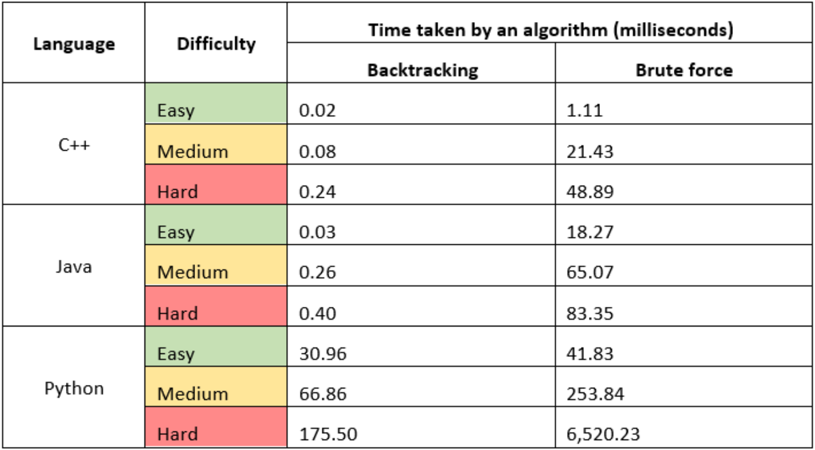
We also wanted to use the OpenCV library to read a sudoku from an image and solve it.

Current Status

- Tested backtracking and brute force algorithm in C++, Java and Python for 100 sudokus of 3 difficulty levels.
- Tested stochastic simulated annealing algorithm and Crook's algorithm in Python - not working for higher difficulty levels.

Current Status

- Results

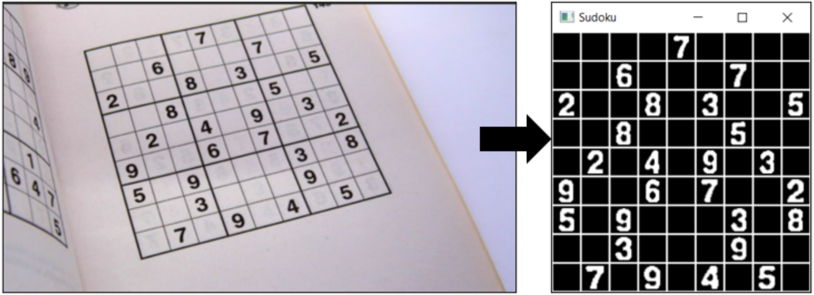


Language	Difficulty	Time taken by an algorithm (milliseconds)	
		Backtracking	Brute force
C++	Easy	0.02	1.11
	Medium	0.08	21.43
	Hard	0.24	48.89
Java	Easy	0.03	18.27
	Medium	0.26	65.07
	Hard	0.40	83.35
Python	Easy	30.96	41.83
	Medium	66.86	253.84
	Hard	175.50	6,520.23

Figure: Average time taken to solve a sudoku (tested 100 puzzles).

Current Status

- Completed image processing of a sudoku.



				7				
		6				7		
2			8		3			5
		8				5		
	2		4		9		3	
9			6		7			2
5		9				3		8
		3				9		
	7		9		4		5	

Figure: Image processing of a sudoku.

Current Status

- Extracted cells from the sudoku.

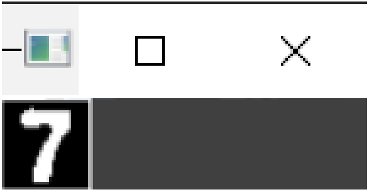
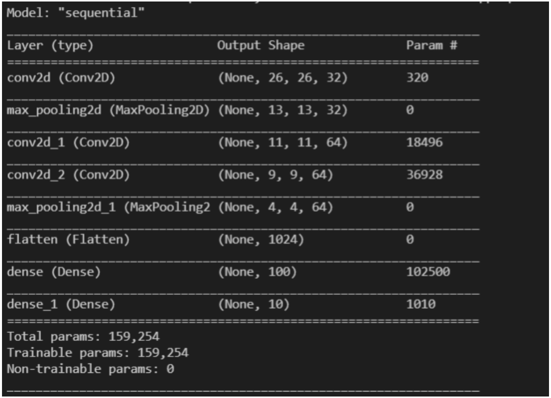


Figure: Extracted cell - row 1 and column 5.

Current Status

- Created a CNN model for predicting the digits in the sudoku using MNIST dataset.

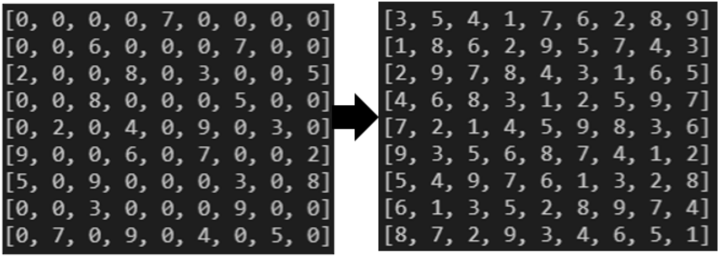


```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 26, 26, 32)        320
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 11, 11, 64)        18496
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 9, 9, 64)          36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 4, 4, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 1024)              0
_________________________________________________________________
dense (Dense)                (None, 100)               102500
_________________________________________________________________
dense_1 (Dense)              (None, 10)                1010
=================================================================
Total params: 159,254
Trainable params: 159,254
Non-trainable params: 0
_________________________________________________________________
```

Figure: CNN model summary.

Current Status

- Identified the sudoku from the image and solved it using backtracking algorithm.



```
[0, 0, 0, 0, 7, 0, 0, 0, 0]
[0, 0, 6, 0, 0, 0, 7, 0, 0]
[2, 0, 0, 8, 0, 3, 0, 0, 5]
[0, 0, 8, 0, 0, 0, 5, 0, 0]
[0, 2, 0, 4, 0, 9, 0, 3, 0]
[9, 0, 0, 6, 0, 7, 0, 0, 2]
[5, 0, 9, 0, 0, 0, 3, 0, 8]
[0, 0, 3, 0, 0, 0, 9, 0, 0]
[0, 7, 0, 9, 0, 4, 0, 5, 0]
```

```
[3, 5, 4, 1, 7, 6, 2, 8, 9]
[1, 8, 6, 2, 9, 5, 7, 4, 3]
[2, 9, 7, 8, 4, 3, 1, 6, 5]
[4, 6, 8, 3, 1, 2, 5, 9, 7]
[7, 2, 1, 4, 5, 9, 8, 3, 6]
[9, 3, 5, 6, 8, 7, 4, 1, 2]
[5, 4, 9, 7, 6, 1, 3, 2, 8]
[6, 1, 3, 5, 2, 8, 9, 7, 4]
[8, 7, 2, 9, 3, 4, 6, 5, 1]
```

Figure: Solved the sudoku from the image

- Agenda for next week: try to solve sudoku using Haskell and Elixir.

Toolchain

- Languages: Python, C++, Java, Haskell, Elixir.
- Libaries used in Python: Numpy, OpenCV and Keras.

Learnings

We learnt to:

- Collaborate using GitLab.
- Write the same algorithm in different languages.
- Generate data in one language and use the data in another language.
- Explain our code, thought processess and ideas to each other.
- Apply the concept of cost function and thermodynamics in simulated annealing.
- Generate 100 sudokus of 3 different difficulty levels.
- Process an image to extract digits of a sudoku.
- Implement neural networks to predict digits of a sudoku from an image.