The Sudoku Project

WEEK 8: 12/09/2021 to 18/09/2021

Ishika De and Yashvi Donga

September 2021

Agenda

- Brief Overview
- Current Status
- Toolchain
- Difficulties
- Learnings

Brief Overview

The goal of this project is to investigate a variety of algorithms (backtracking, brute force, stochastic search, depth first search) that are capable of solving standard Sudoku puzzles, of ranging difficulties, in order to learn more about Sudoku solving techniques.

We also wanted to create the sudoku solver using OpenCV that will read a puzzle from an image and solve it. We plan on using OpenCV for multiple programming languages.

Current Status

- Implemented Backtracking and Brute Force Algorithm to solve a Sudoku in C++, Java and Python.
- Tested 100 Sudokus of different difficulties for Backtracking and Brute Force Algorithm in C++, Java and Python.
- Generated a graph to compare the algorithms and languages with respect to time taken.
- Tried implementing Stochastic Simulated Annealing Algorithm in Python. (Not working for all cases)
- Tried implementing Crook's Algorithm in Python. (Not working for all cases)

Current Status

- RESULTS

LANGUAGE	DIFFICULTY	Time taken by Algorithm (milliseconds)	
		Backtracking	Brute force
C++	Easy	0.0230	1.1129
	Medium	0.0795	21.4268
	Hard	0.2391	48.8923
Java	Easy	0.0259	18.2688
	Medium	0.2556	65.0726
	Hard	0.3961	83.3507
Python	Easy	30.9598	41.8319
	Medium	66.8602	253.8408
	Hard	175.5002	6520.2295

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

Current Status

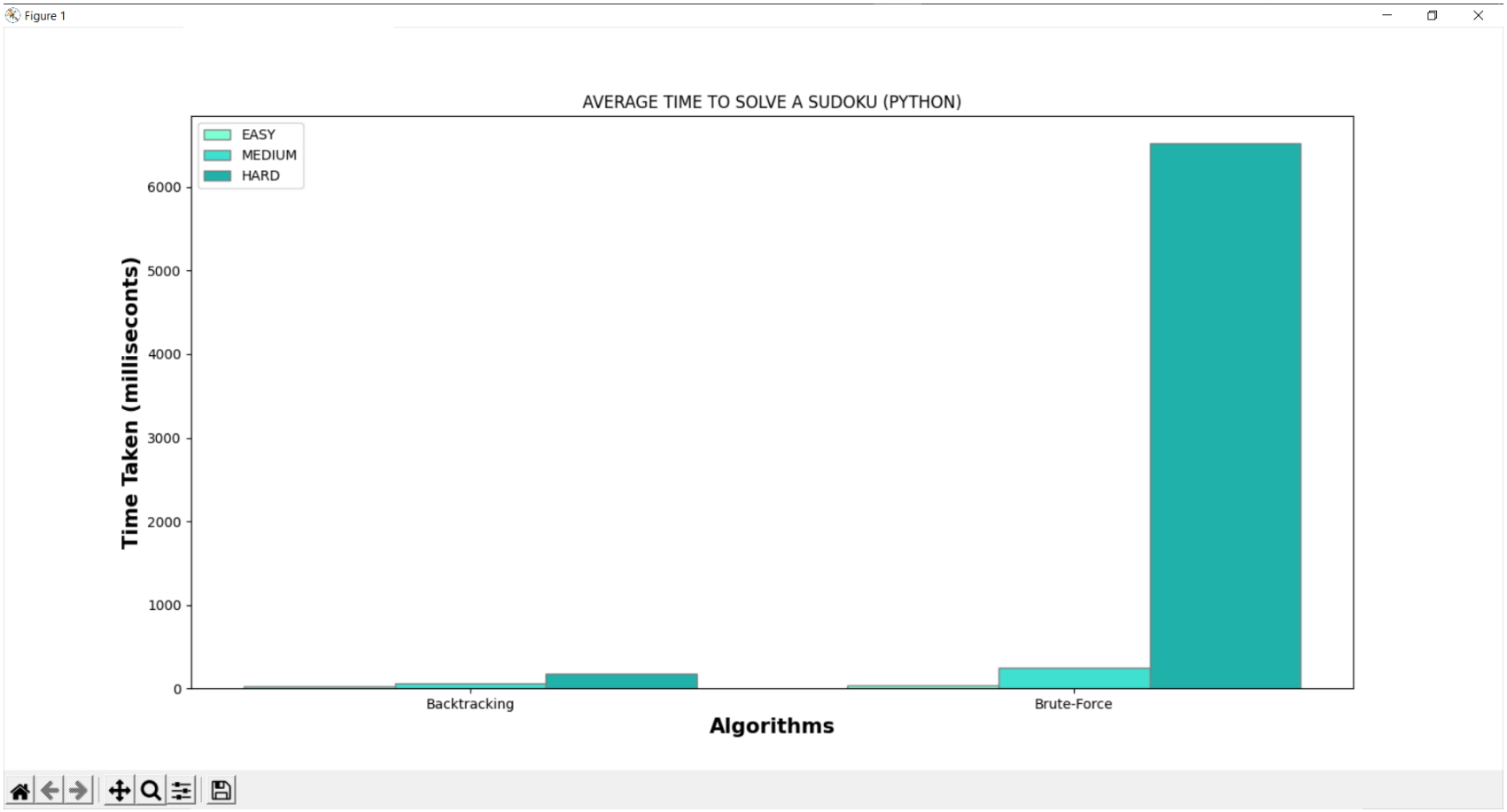

Figure: Results: Average time taken to solve a sudoku in Python.

Current Status

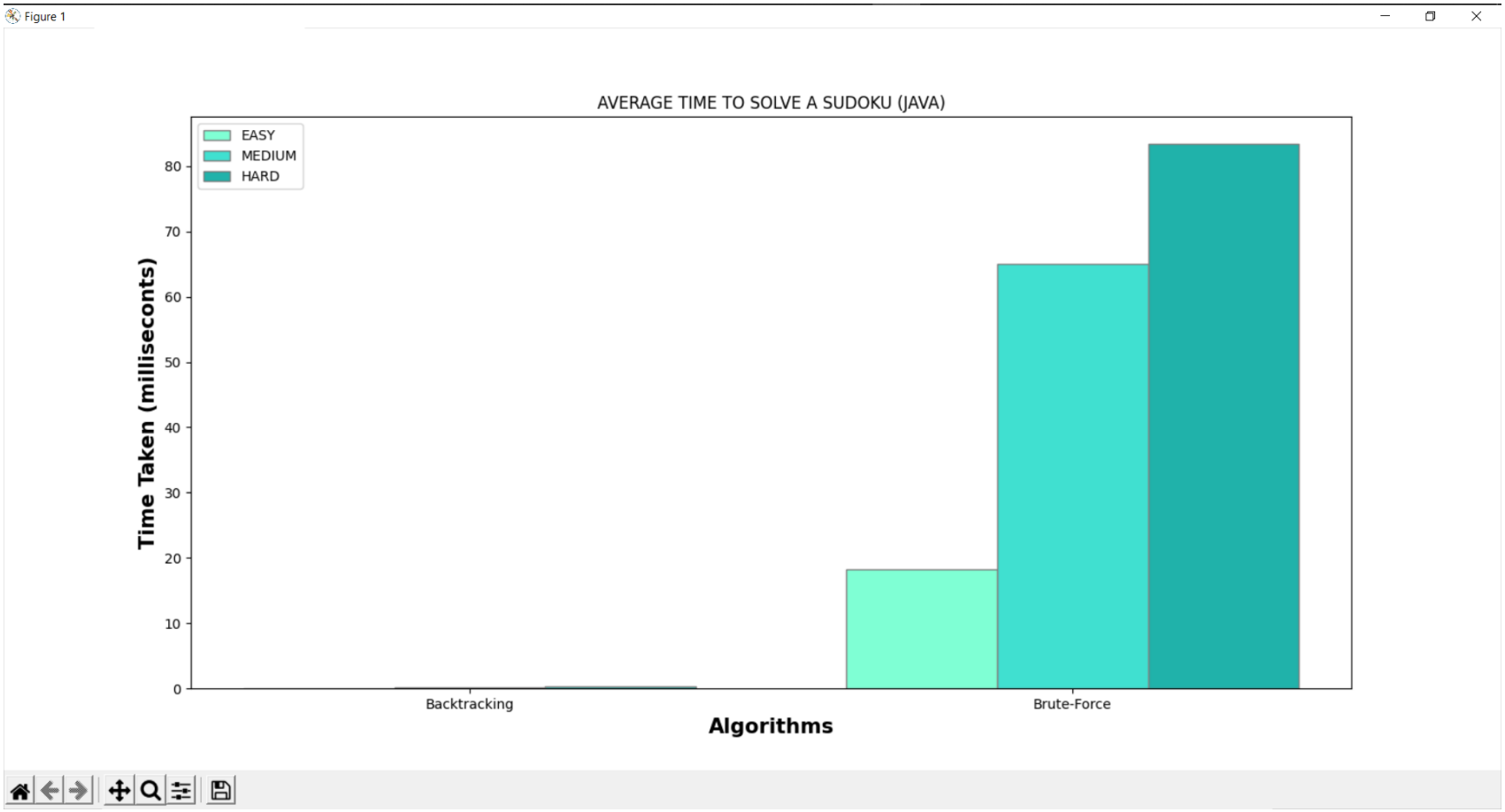

Figure: Results: Average time taken to solve a sudoku in Java.

Current Status

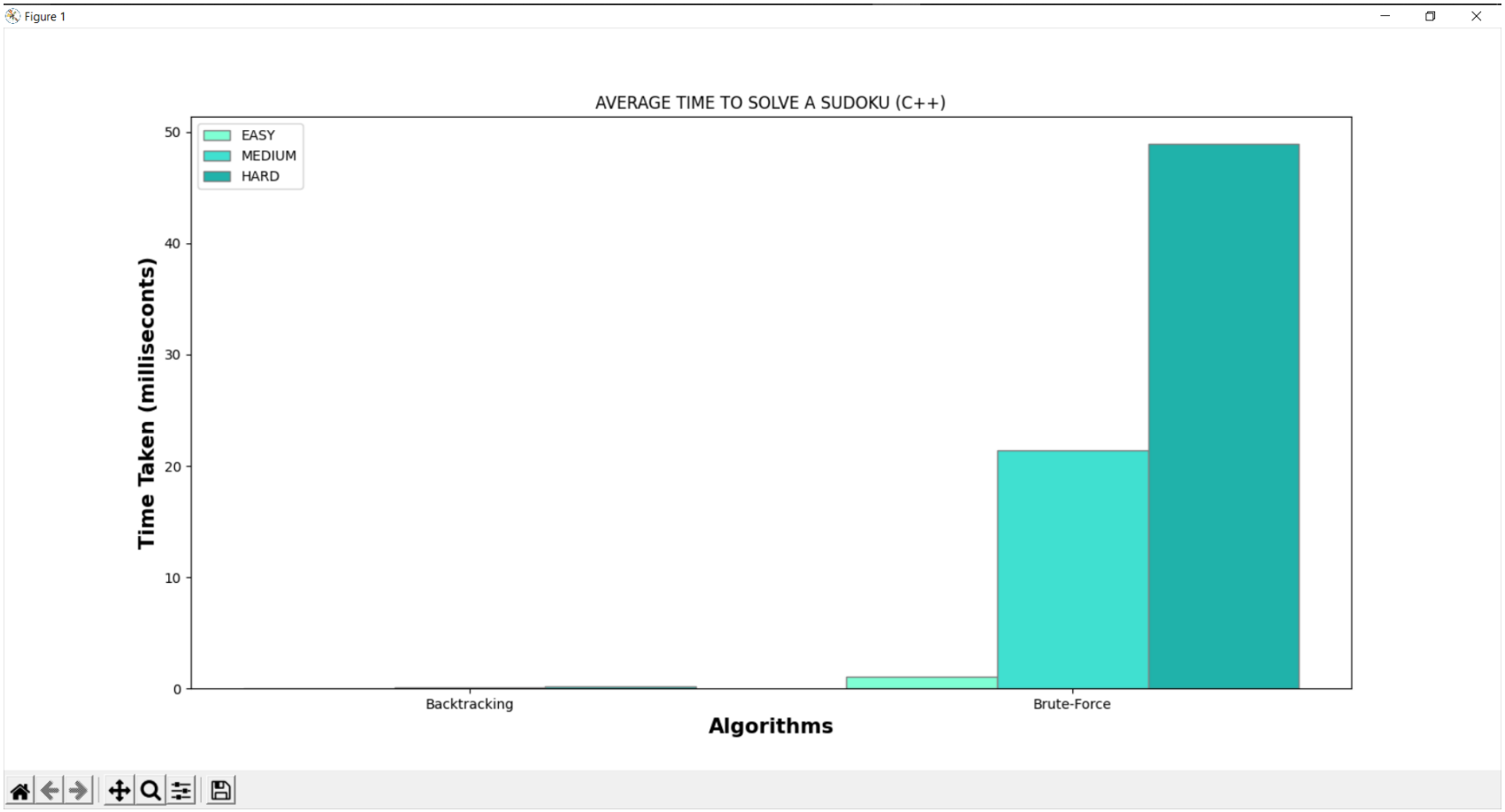

Figure: Results: Average time taken to solve a sudoku in C++.

Current Status

- Completed image processing of a sudoku puzzle image.

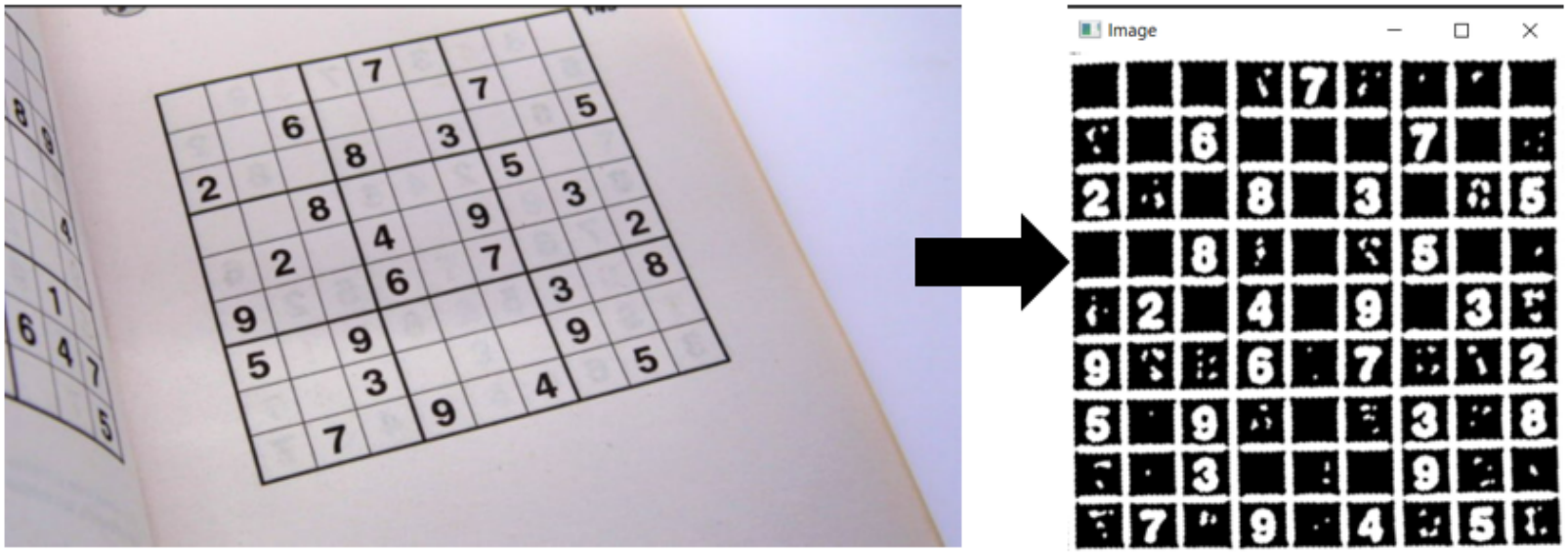

Figure: Image processing of a random sudoku puzzle image.

Current Status

- Completed Extracting cells from the sudoku.

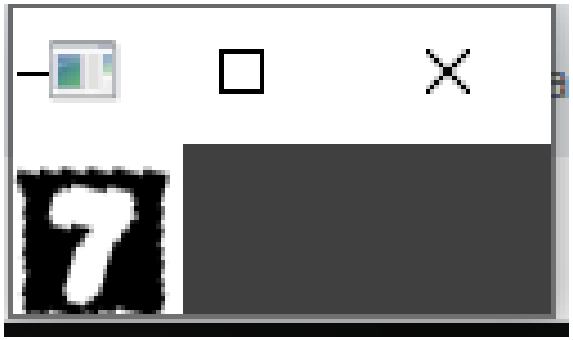

Figure: Extracting Cell - Row 1 and Column 5.

- Read and learnt concepts of neural networks, machine learning and image processing.

Current Status

- Created a CNN Model for predicting the Numbers 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.

- The numbers being predicted by the CNN model are wrong.

Toolchain

- Languages: Python,C++, Java, Haskell, Elixir.
- Libaries Used: time module (in Python, C++ and Java) and matplotlib(python)
- Open CV
- Keras - for Creating Model of the MNIST Dataset.

Difficulties

- The backtracking algorithm took some time to implement because we had few challenges implementing the recursive function.
- Understanding Stochastic Simulated Annealing was difficult. The resources for this algorithm were quite less.
- For stochastic and crook's algorithm approach, the sudoku solver does not work for all cases. This requires a little bit of work.
- Explaining each other's ideas to each other took a while.
- Generating 100 sudokues of 3 different difficulty levels was a challenge because it was taling lot of time to generate the puzzle and we had to optimize the code.
- Understanding the concepts of Image Processing and neural networks took us a while.
- Extracting and Predicting the digits.

Learnings

- Learning about the implementation of different algorithms to solve a Sudoku.
- Collaboration and understanding git commands.
- Explaining our code, thought process and ideas to each other.
- Understanding the concepts implemented in Stochastic algorithm using Simulated Annealing.
- Generating data in one language, importing and using the data in another language.
- Learnt about neural networks and its applications.
- Processed and extracted digits from Sudoku using OpenCV Library in Python.
- Using the Keras library and creating a CNN Model for predicting Digits.