“Hand Drawn” Sudoku Solver

Team: Aiden Sullivan , Sam Avis, Jimmy Schaffer, Dolan Clahan, Spencer Webb

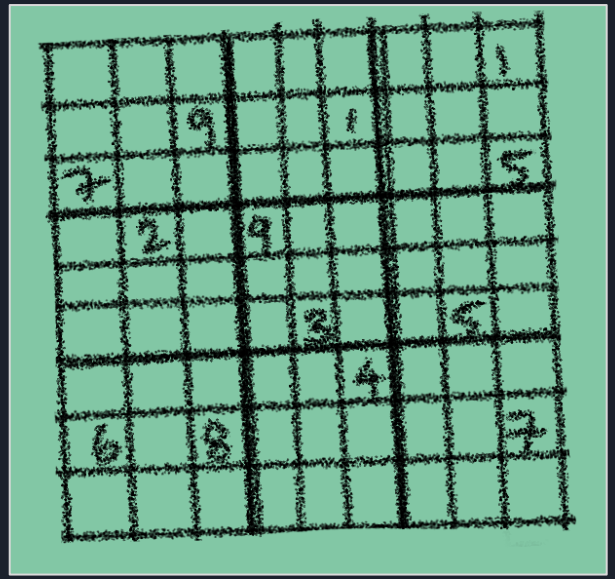
Define problem

Standard 9x9 Sudoku puzzles

Use computer vision to read and solve “hand drawn” sudoku problems.

Scored based on number of completely correct solutions matched

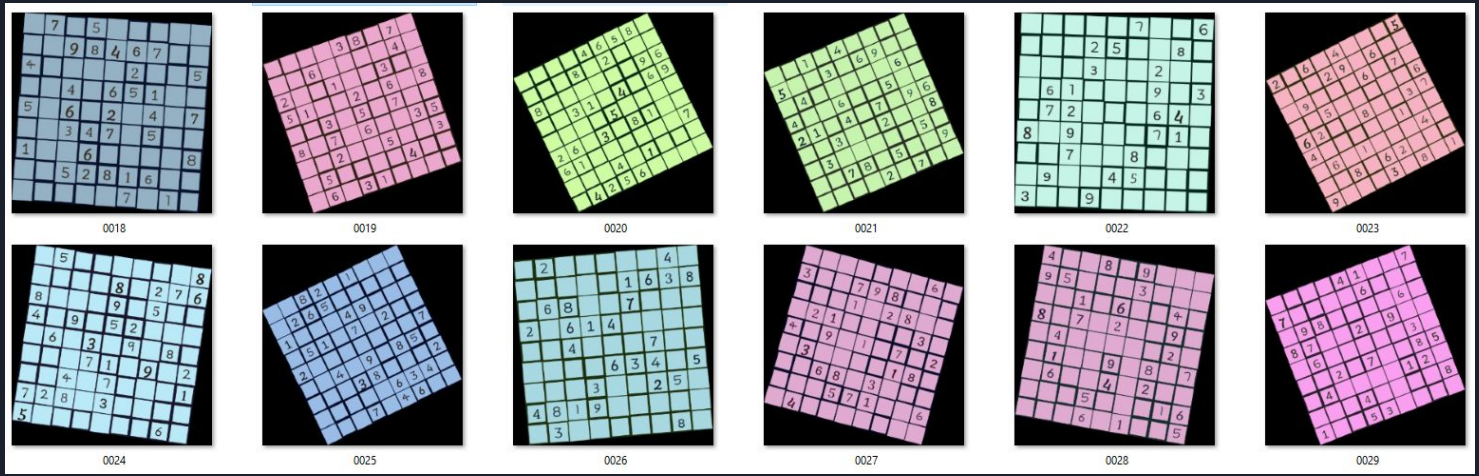
Returns solution in row major form
(971683245562479831483152679854267193219538764637914582148795326396821457725346918)



[Alcrowd | SUDOKU | Challenges]

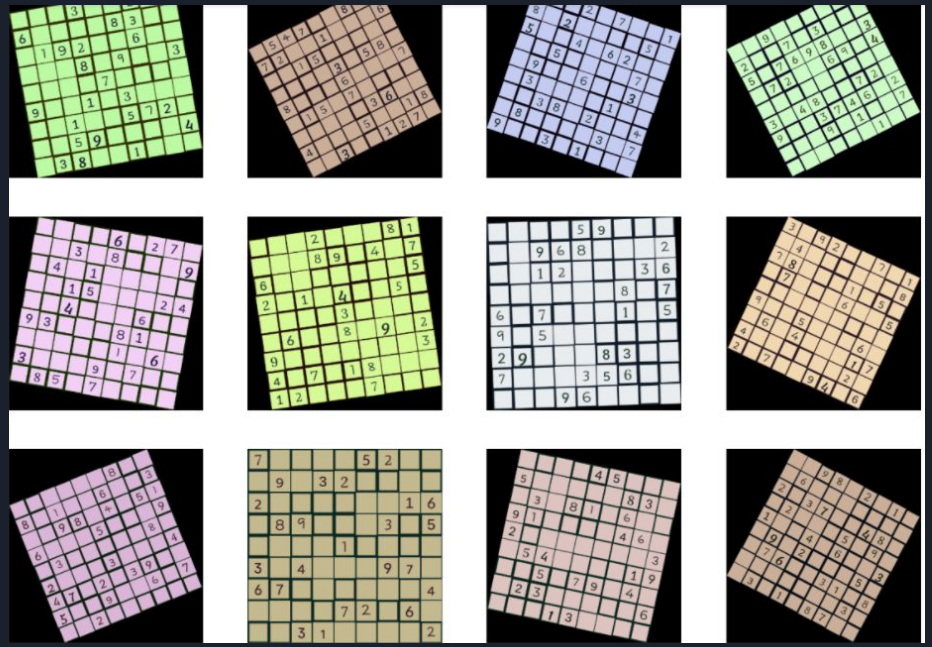
Drawings:

- Drawings each have a different
  - Color
  - Rotation (always vertically oriented)
  - Line widths within the drawing
- All drawings are (300,300) pixels but due to rotation the input size varies



Available Data

- Dataset of 10001 images
  - 5000 training puzzles
  - 5001 test puzzles
- Training puzzles have labeled solutions in row major form
- Starting file to get the basics



Proposed Solution

Split the problem into 4 subparts

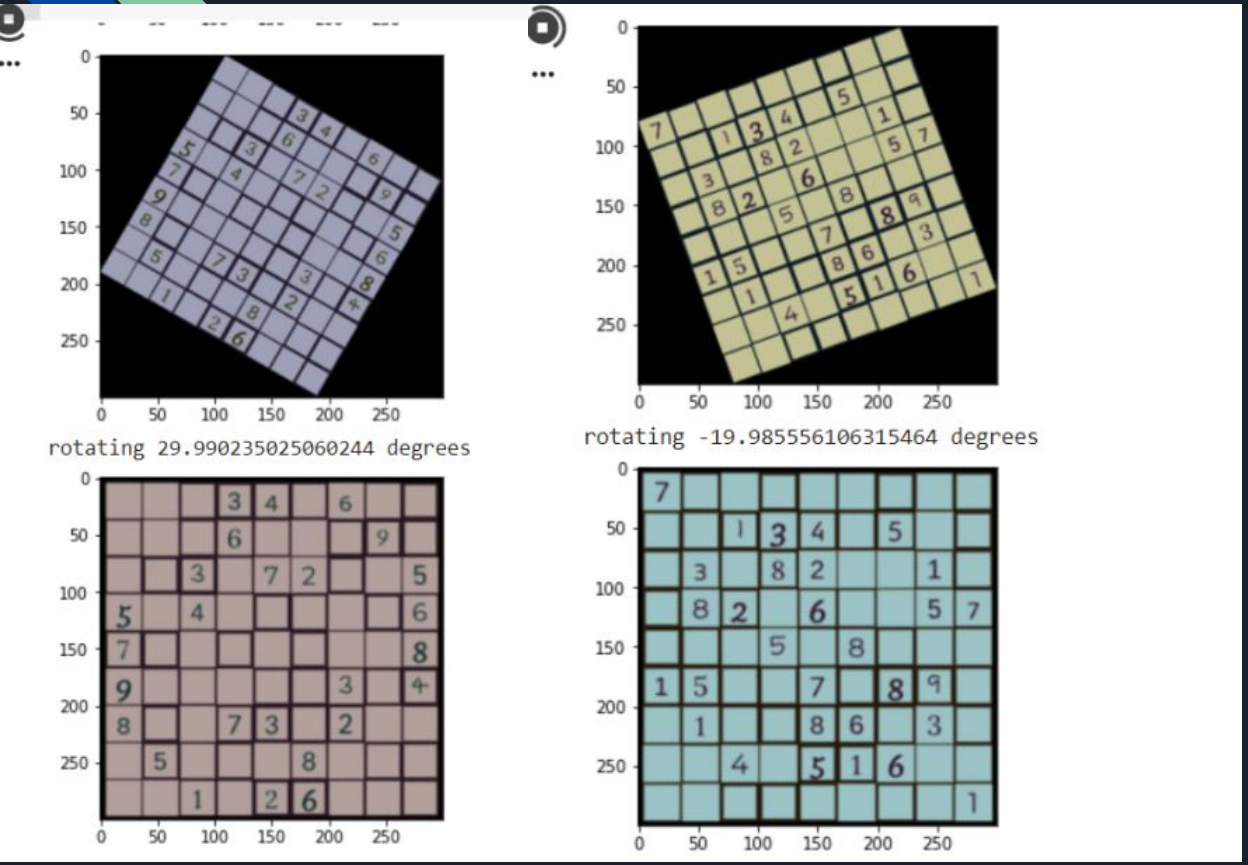
1. Normalize image data: Only take into account numbers and lines ignoring color
2. Neural Network to straighten and crop puzzle to original (300x300) pixel image
3. Neural Network applied to each cell to extract number data
4. Solve puzzle based on gathered data

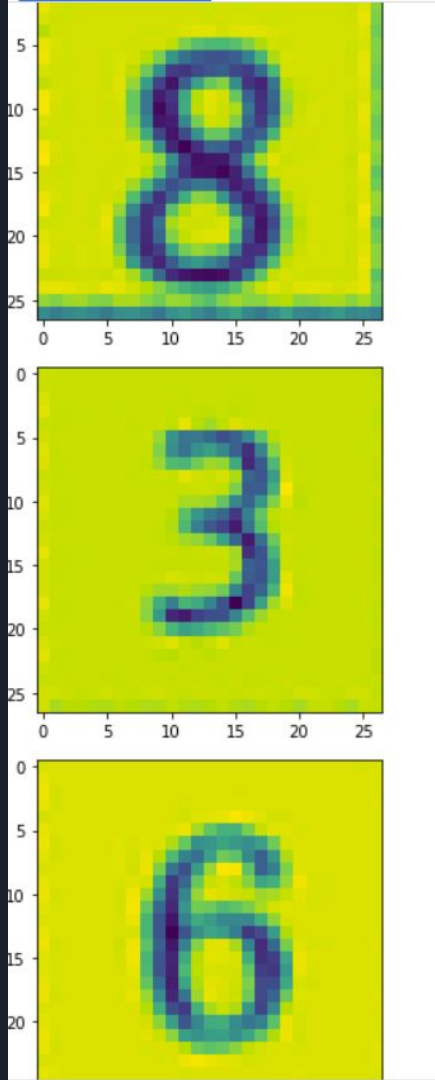
UPDATE 8/20

We have been able to properly implement:

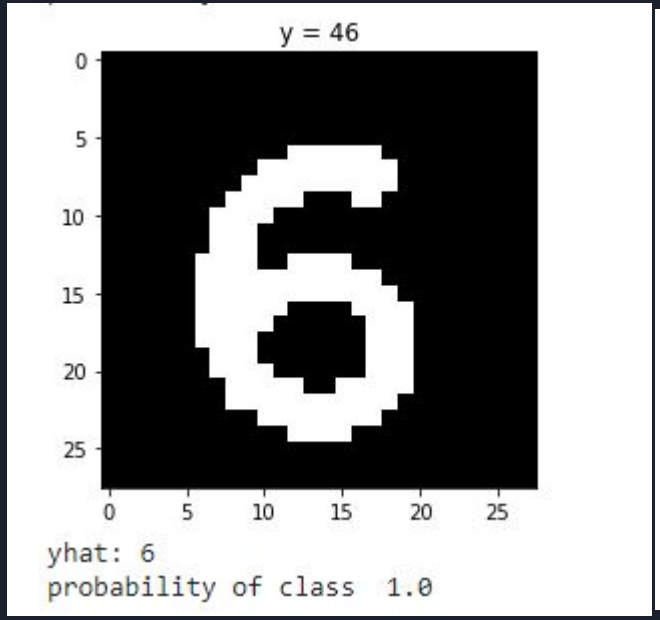
- Rotate the image to its intended 300x300 pixel vertical sudoku puzzle
  - Multiple solutions found, optimized from ~30 seconds to <1 second for rotation.
- Crop the individual cells, accounting for differences in line thickness and number centering.
- Determine whether or not a given cell from a sudoku board is empty or contains some number by binarizing the tensor image and scanning for the presence of pixels in a given cell.
- Training a Neural Network on MNIST dataset for digit recognition.
- Reading the numbers from the individual cells with pre-trained model and formatting into solvable Sudoku() object.
- Training our own model using digits from non-empty cells from sudoku boards. Validating based on training data answers.
- Solved Sudoku puzzle! Accuracy? So-so. Any one mis-read digit and the sudoku solver is left with an unsolvable board, high margin for error.
- We were always aware of how integrating the many working parts of this project would be a challenge, and can confirm that that is the case.

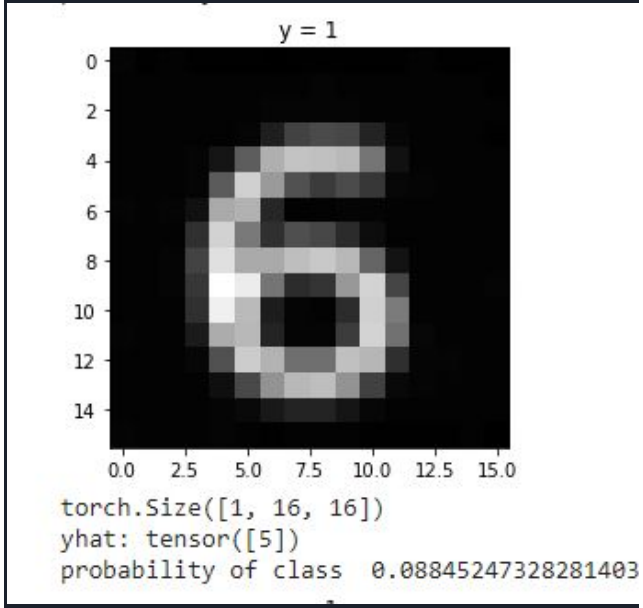
Rotation and Cell Retrieval

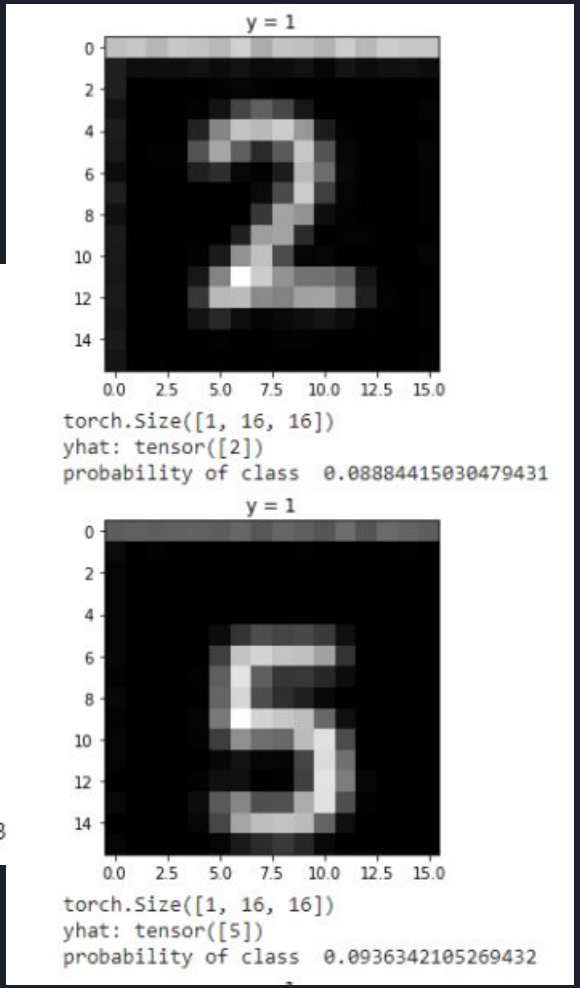




Digit Recognition



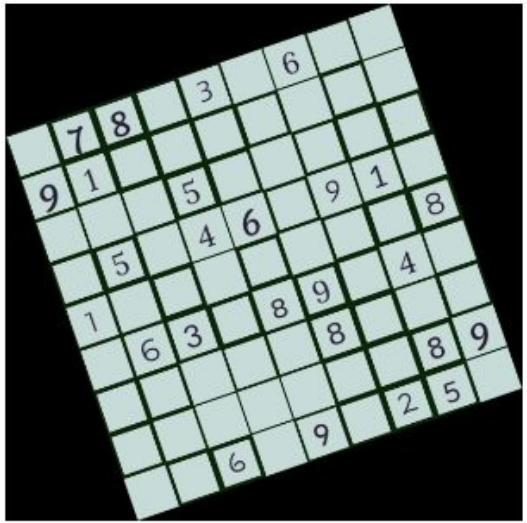


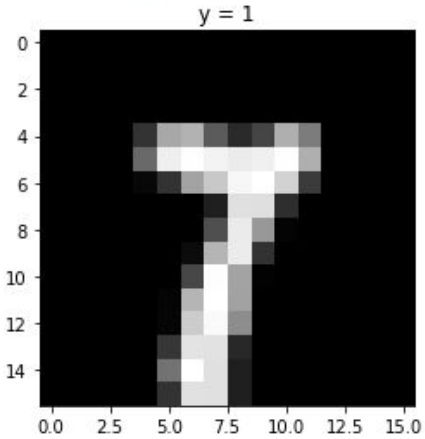


And misrecognition

Rotating, Cropping, and Recognizing digits of Sudoku puzzle

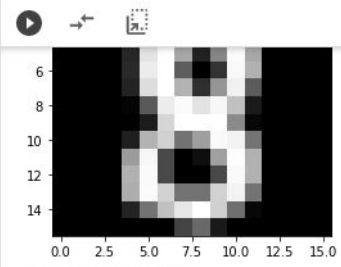
First 7 digits, 72% accuracy for digit recognition for this puzzle for this iteration of NN digit recognition model



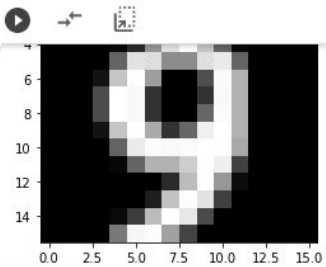


yhat: tensor([7])
probability of class 0.1424865871667862

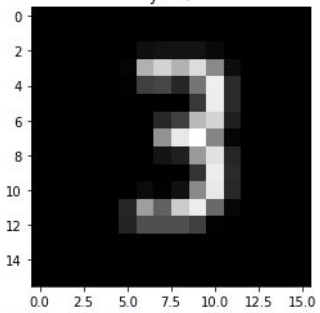
y = 2



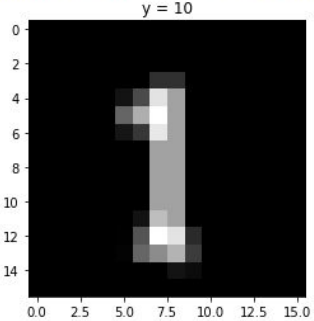
yhat: tensor([8])
probability of class 0.121027141809463



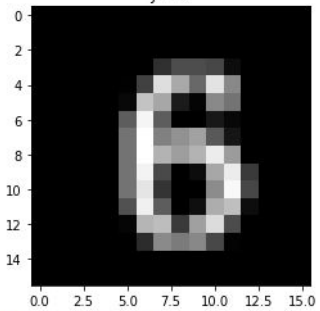
yhat: tensor([9])
probability of class 0.1368194818496704



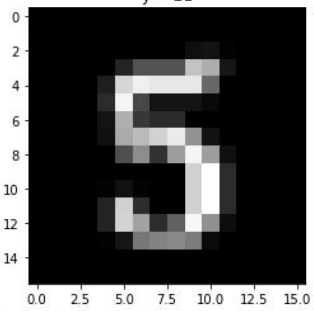
yhat: tensor([3])
probability of class 0.137482643127441



yhat: tensor([1])
probability of class 0.13772441446781158



yhat: tensor([5])
probability of class 0.140339225530624



yhat: tensor([5])
probability of class 0.1339673548936844