פותר סודוקו דיגיטלי

<u>פרויקט בעקרונות שפות תכנות</u>

מרצה: אדיר סולומון

29/12/20

<u>מגישים:</u>

מיר האוזמן 316587484

פלג אברהם 316472679

תוכן עניינים:

מבנה הדוח

תיאור הפרויקט 3
סקירת התחום 4
סקירת ספרות..... 5
סקירת הכלים הטכניים..... 9
הוראות התקנה 10
מאגרי המידע..... 11
שימוש פונקציונלי 12
פונקציות עיקריות במערכת..... 12
השימוש בכלי kivy..... 15
השימוש בכלי opencv..... 21
תוצאות 24
תוצאות השימוש בכלי kivy..... 24
מסך ראשי:..... 26
About :..... 27
Recommends :..... 28
מסך בחירה:..... 29
חלון מעבר למשחק לאחר בחירת תמונה:..... 30
מסך משחק:..... 31
חלון סיום לאחר ניצחון:..... 32
תוצאות השימוש בכלים לעיבוד תמונה..... 33
תוצאות השימוש בכלים ללמידה עמוקה..... 35
ניתוח המידע..... 35
בניית המודל..... 36
אימון המודל..... 37
מסקנות 39
כלי kivy..... 39
הספרייה openCV 39
כלי keras..... 40

תיאור הפרויקט

מהי המשימה ?

לפענח תמונת סודוקו וליצור ממשק נאה דרכו יוכל המשתמש להמשיך לפתור את הסודוקו,

כאשר ביכולתו לגלות את הפתרון בכל שלב במשחק.

המשתמש מעלה תמונה המתארת סודוקו לא פתור או פתור חלקית,

המערכת מפענחת את התמונה ומזהה את הטבלה התאים והספרות,

היא מעבירה את המשתמש למשחק אינטראקטיבי,

ביכולת המשתמש למלא את הטבלה, ולהציג את הפתרון עבור הטבלה הנוכחית,

עבור קלטים שגויים המערכת מציגה למשתמש תגובה ויזואלית הולמת.

מהי המוטיבציה לשימוש בתוכנית ?

מה יותר קלאסי מסודוקו ? המשחק הכי פופולרי שידעה האנושות עוד מימי קדם

כמה פעמים נתקלתם בעיתון בסודוקו נושא פרסים, ואמרתם לעצמכם לו רק היה לי סבלנות לפתור אותו עד הסוף מתישהו הייתי זוכה בפרס ! אז החלום נהפך מציאותי בעזרתנו.

בנוסף, פותרי הסודוקו נחלקים לכמה רמות: מתחילים, בינוניים ומומחים.

מעבר לנוחות שיש בפתירת הסודוקו דרך המחשב, הוויזואליות והפידבק הישיר המגיע עם הזנת המספרים, השייכת לכלל רמות השחקנים,

המערכת שלנו מספקת למשתמש לקבל פתרון חלקי עד מלא עבור הלוח הקיים.

המתחילים צריכים את הסיוע, כיוון שעדיין אינם מתמצאים במשחק, לכן התוכנית מאוד תועיל עבורם.

הבינוניים יכולים להיעזר בגילוי של ספרות בודדות מתוך הלוח, היכן שייתקעו זמן רב.

המומחים, יוכלו לערוך תחרות עם חבריהם, כאשר הם ימלאו את הסודוקו בדף וחבריהם ימלאו במחשב עם מספר רמזים נתון, וכך יבחנו מי ידו על העליונה, המומחה או הבינוני בסיוע המחשב.

מהי המוטיבציה לפתח מערכת כזו ?

התעסקות עם למידה עמוקה, עיבוד תמונה, פיתוח אפליקציה, ואלגוריתמיקה.

זיהוי ספרות ע"י פרדיקציה של מודל בנוי רשת נוירונים,

עיבוד התמונה ע"י ספריית opencv,

פיתוח אפליקציה ע"י ספריית kivy המדהימה,

וכתיבת אלגוריתם לפתירת לוח המשחק.

אפשר לבקש יותר ? חלומו של כל מתכנת, בפרט פייתונאי :)

סקירת התחום

הפרויקט אותו אנו עושים מגיע כרעיון חדש שלא נעשה בעבר, לפי ערך השגתנו.

חלקים ממנו נעשו לפנינו ע"י רבים וטובים, אך לא באותה פלטפורמה אותה אנו מפתחים.

לפי חקר אינטרנטי, מצאנו שאת הפרויקטים שנעשו בהקשר של סודוקו ופייתון ניתן לסווג לשני נושאים.

הראשון, ניתוח תמונה והדפסת התוצאות / הצגת התוצאות על גבה, מרחיקי הלכת תמכו בזאת בזמן ריצה.

השני, פיתוח משחק באופן עצמאי, כאשר הפרויקטים שנעשו היו ברמות שונות של מורכבות. החל מתכונית פשוטה כמו פתרון רגיל של סודוקו על גבי דף, וכלה בתוכנית שכוללת רמזים ויכולת חזרה אחורה במהלכים.

בפרויקט שלנו אנו כוללים את שני החלקים יחדיו, אנו מפענחים את התמונה כמו החלק הראשון, ויוצרים ממשק עבור המשך המשחק במחשב כמו בחלק השני. וכוללים כמובן את היכולת לפתור את הסודוקו ע"י עזרה מהמחשב.

בקישורים להלן נציג חלק מתוך מגוון מקורות הכוללים מדריכים ופרויקטים לדוגמא משני הנושאים הנ"ל.

המכנה המשותף לכל המדריכים בחלק הראשון הוא השימוש בחבילת opencv לפענוח הטבלה והשימוש בלמידת מכונה עבור חיזוי הספרות

בחלק השני הוא מצאנו יותר מימושים העושים שימוש בחבילת kivy לפיתוח הממשק, והיו פרויקטים מעטים שהשתמשו ב tkinter.

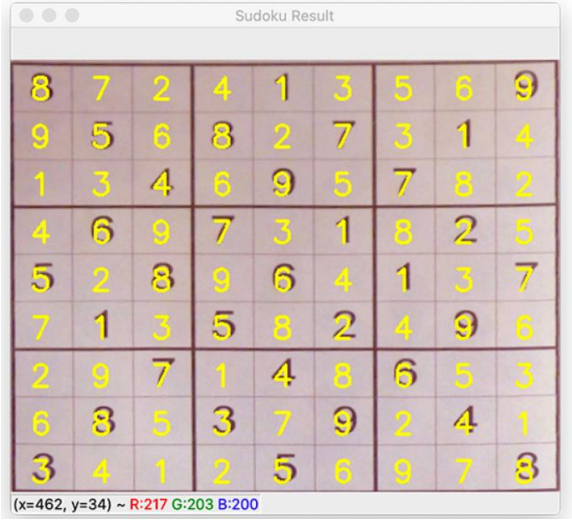
כתוצאה מכך בחרנו להעמיק את הסקירה בכלים הטכניים דווקא על opencv , כלים בלמידה עמוקה ו kivy .

סקירת ספרות

נציג חלק מהאתרים שסקרנו אודות פרויקטים שנעשו בנושאים הנ"ל

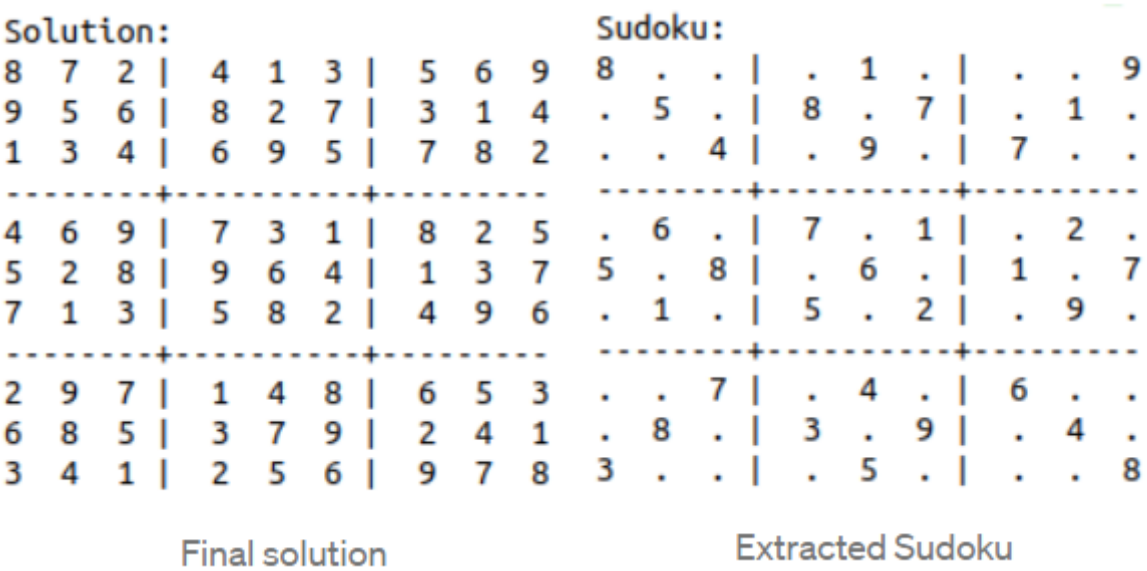
1. פענוח התמונה והדפסת התוצאות / הצגת תוצאות על גבה:

https://www.pyimagesearch.com/2020/08/10/opencv-sudoku-solver-and-ocr/



https://aakashjhawar.medium.com/sudoku-solver-using-opencv-and-dl-part-1-490f08701179

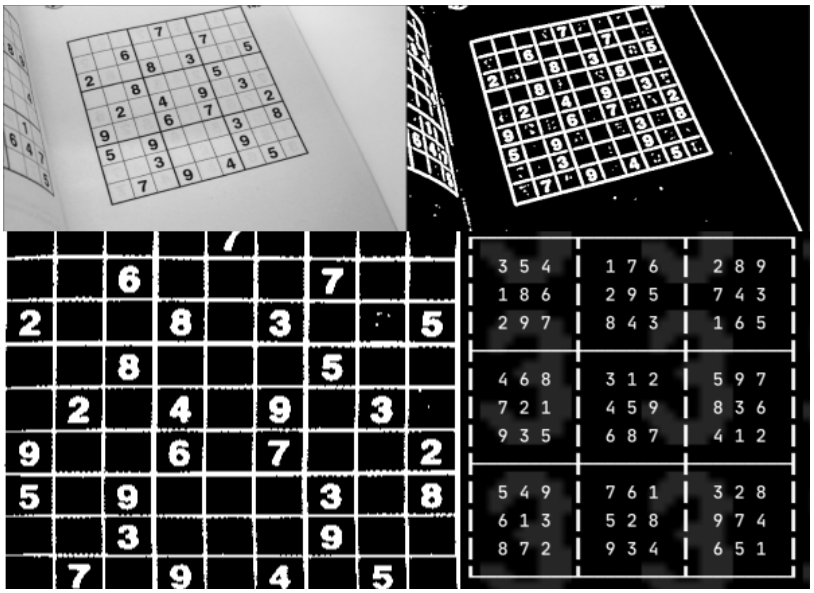
https://aakashjhawar.medium.com/sudoku-solver-using-opencv-and-dl-part-2-bbe0e6ac87c5



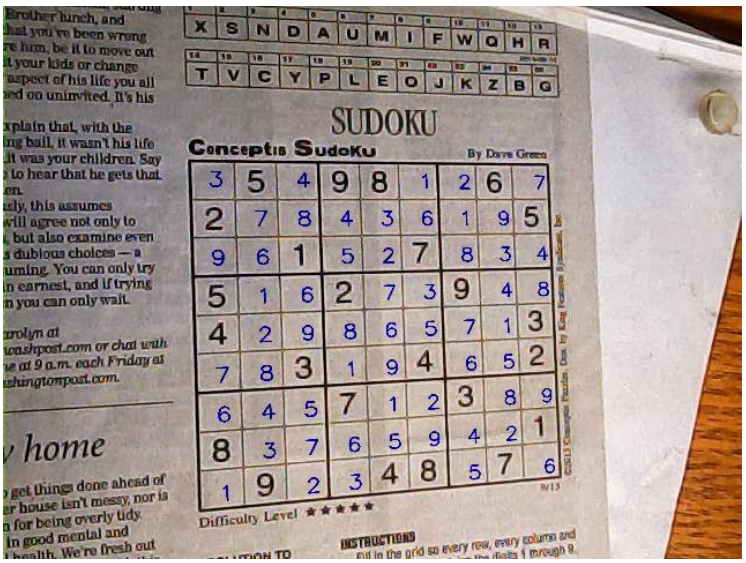
https://becominghuman.ai/image-processing-sudokuai-opencv-45380715a629

https://becominghuman.ai/sudoku-and-cell-extraction-sudokuai-opencv-38b603066066

https://becominghuman.ai/part-3-solving-the-sudoku-ai-solver-13f64a090922



https://towardsdatascience.com/open-cv-based-sudoku-solver-powered-by-rust-df256653d5b3



/https://www.data-stats.com/sudoku-solver-using-opencv

```
Solution:                                   Sudoku:
8  7  2 |  4  1  3 |  5  6  9   8  .  . |  .  1  . |  .  .  9
9  5  6 |  8  2  7 |  3  1  4   .  5  . |  8  .  7 |  .  1  .
1  3  4 |  6  9  5 |  7  8  2   .  .  4 |  .  9  . |  7  .  .
--------+----------+---------  --------+----------+---------
4  6  9 |  7  3  1 |  8  2  5   .  6  . |  7  .  1 |  .  2  .
5  2  8 |  9  6  4 |  1  3  7   5  .  8 |  .  6  . |  1  .  7
7  1  3 |  5  8  2 |  4  9  6   .  1  . |  5  .  2 |  .  9  .
--------+----------+---------  --------+----------+---------
2  9  7 |  1  4  8 |  6  5  3   .  .  7 |  .  4  . |  6  .  .
6  8  5 |  3  7  9 |  2  4  1   .  8  . |  3  .  9 |  .  4  .
3  4  1 |  2  5  6 |  9  7  8   3  .  . |  .  5  . |  .  .  8
```

https://levelup.gitconnected.com/solving-a-sudoku-puzzle-using-deep-learning-and-backtracking-algorithm-c6cef475ae3

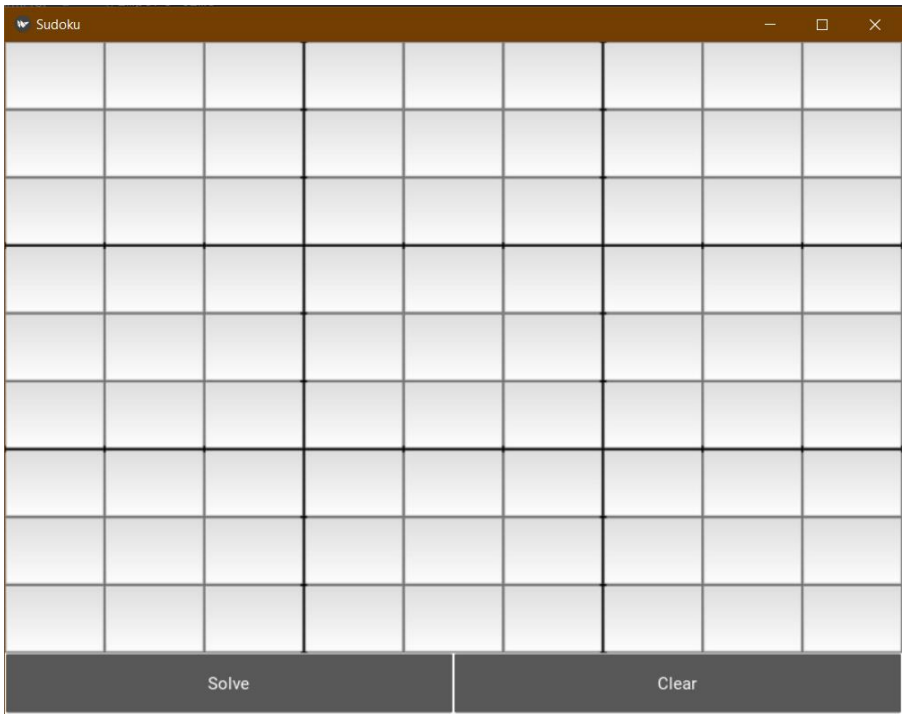
5	7	6	8	1	2	4	9	3
8	1	2	4	3	4	5	6	7
9	3	4	5	7	6	1	8	2
4	9	1	3	5	8	2	7	6
7	5	8	6	2	4	9	3	1
6	2	3	1	9	7	8	5	4
1	6	9	7	4	5	3	2	8
3	8	5	2	6	1	7	4	9
2	4	7	9	8	3	6	1	5

	1	2		3	4	5	6	7
	3	4	5		6	1	8	2
		1		5	8	2		6
		8	6					1
	2				7		5	
		3	7		5		2	8
	8			6		7		
2		7		8	3	6	1	5

2. משחק סודוקו:

מימוש פשוט ב kivy :

https://github.com/uraza/kivy-sudoku-app/blob/master/main.py

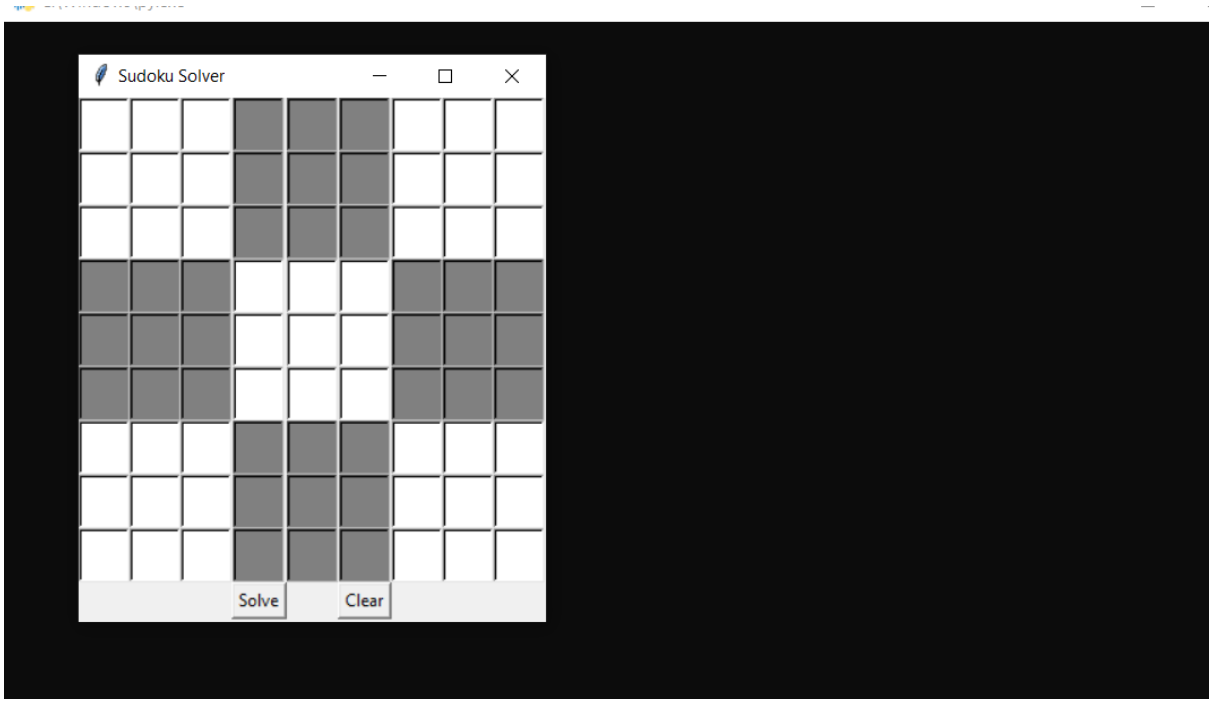


מימוש מורכב הכולל היסטוריה, צעדים חוזרים ורמזים ב kivy:

https://github.com/gagahan/sudoku

מימוש נאה ב tkinter :

/https://code-projects.org/sudoku-solver-in-python-with-source-code



סקירת הכלים הטכניים

עפ"י סקירה של מדריכים ופרויקטים שנעשו בעבר אשר חלקם הוצג לעיל, ראינו כי הכלים העיקריים של פייתון בהם עתיד להיעשות שימוש הם חבילות opencv עבור עיבוד התמונה ו kivy עבור יצירת הממשק.

ובנוסף שימוש בספריות keras , mathplotlib , numpy וכיו"ב עבור בניית המודל לזיהוי הספרות

חקרנו עליהם דרך האתרים הבאים :

Opencv:

https://www.geeksforgeeks.org/opencv-python-tutorial/

https://docs.opencv.org/master/d9/df8/tutorial_root.html

kivy:

מצגת שיעור 5

https://kivy.org/doc/stable/guide/basic.html

https://www.javatpoint.com/kivy

https://www.youtube.com/playlist?list=PLzMcBGfZo4-kSJVMyYeOQ8CXJ3z1k7gHn

deep learning tools:

קורס בסדנא מעשית בלמידה עמוקה בראשות נתנאל שמעוני

הוראות התקנה

מערכת הפעלה וינדוס 10

החבילות שהשתמשנו בהן :

frontend.py

```
import math
import copy

from kivy.clock import Clock
from kivy.properties import ObjectProperty
from kivy.app import App

from kivy.uix.button import Button
from kivy.uix.gridlayout import GridLayout
from kivy.uix.screenmanager import Screen, ScreenManager
from kivy.uix.textinput import TextInput
from kivy.uix.image import Image
from kivy.uix.popup import Popup
from kivy.uix.floatlayout import FloatLayout
from kivy.uix.widget import Widget

from kivy.core.window import Window

from solver import solve_sudoku
from scanner import get_board
```

digit_classifier.py

```
import argparse
import numpy as np

import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, accuracy_score
import seaborn as sns

from tensorflow.keras.models import Sequential
from tensorflow.keras.datasets import mnist
from tensorflow.keras.layers import Dense, Conv2D, Flatten, MaxPool2D,
Dropout
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.callbacks import TensorBoard, EarlyStopping,
ModelCheckpoint, ReduceLROnPlateau
from sklearn.preprocessing import LabelBinarizer
```

scanner.py

```
import cv2 as cv
import numpy as np
from imutils.perspective import four_point_transform
from skimage.segmentation import clear_border
import tensorflow as tf
from skimage.transform import resize
```

מאגרי המידע

מאגר המידע עבור זיהוי הספרות הוא mnist השייך ל keras.

המאגר כולל ספרות כתב יד. עבור האימון ישנם 60,000 דוגמאות, עבור הטסט ישנם 10,000 דוגמאות.

בחרנו במאגר כיוון שמתוך סקירת הספרות מצאנו שכמעט כל המימושים, אם לא כולם, אימנו את המודל שלהם על מאגר זה.

הוא ידוע בדיוק הגבוה שמשיגים ממנו ובגודל ומגוון הדוגמאות שהוא כולל.

בנוסף, הלוח יכלול גם פתרון חלקי של המשתמש שהמערכת תצטרך לפענח, לכן חלק בלתי נמנע יהיה לפענח ספרות של כתב יד.

על מנת ליבא את המאגר יש לכתוב:

```
from tensorflow.keras.datasets import mnist
print("[INFO] get data from MNIST...")
(X_train, y_train), (X_test, y_test) = mnist.load_data()
```

פשוט וקל.

ניתוחים סטטיסטים והליך בניית הרשת ואימונה על סמך נתוני המאגר ניתן למצוא לקמן, תחת הכותרת "בנייה ואימון רשת נוירונים לזיהוי ספרות".

שימוש פונקציונלי

פונקציות עיקריות במערכת

frontend.py

```
class MainWindow(Screen):
    """
    The main screen
    Purpose: explain the user about the application
             and navigate him to the selection screen
    """

    def show_popup_about(self):
        """
        open popup with content about the sudoku game
        """


    def show_popup_introductions(self):
        """
        open popup with content about the sudoku game
        """
```

```python
def prepare_game(image_source):
    """
    with path to the image, the function:
    1. scan and get the board
    2. create board to solve on it
    3. create the board of the final sulotion
    :param image_source: path to the image that selected
    """
class SelectionWindow(Screen, GridLayout):

    def selected(self, filename):
        """
        get the image selected
        :param filename: image file
        :return:
        """

    def update_filechooser_font(self, *args):
        """
        update the font of the file names in the file chooser
        :param args:
        :return:
        """

    def update_file_list_entry(self, file_chooser, file_list_entry, *args):
        """
        update the font in the file chooser and support in hebrew file
names
        :param file_chooser:
        :param file_list_entry:
        :param args:
        :return:
        """
```

```python
class GameWindow(Screen, Widget):
    """
    The Game screen
    Purpose: play from the image that we get from the selection screen
             and show the solution or end the game
    """
```

```python
class WindowManager(ScreenManager):
    """
    Manager the screens
    """
```

```python
class Cell(TextInput):
    """
    Cell of one piece of the sudoku grid
    """

    def update_num(self):
        """
        update the number in the boards like the number of the cell.
        react according to rules of the game.
        """
```

```
    def update_when_discovered(self, instance, text):
        """
        in solve mode on focus in cell, its update the value of the cell
according to the solution
        """
```

```
class SolveBtnGrid(GridLayout):
    """
    contain buttons of solve mode and end game.
    """

    def enter_solve_mode(self):
        """
        change the state of the game to permit discover the user the
answer, and update the solution_board
        """

    def exit_solve_mode(self):
        """
        change the state of the game to prohibit discover the user the
answer
        """
```

```
def is_valid(row, col, num):
    """
    check if can put num in the board
    :param row: row in board
    :param col: col in board
    :param num: num value
    :return: true if there is no same number in row and col and the square
of num
    """
```

```
def is_board_complete(current_board):
    """
    check if current_board is complete like the correct solution
    :param current_board: board
    :return: true if the board is complete with the solution
    """
```

```
def show_victory_popup():
    """
    open popup with victory message
    """
```

solver.py

```
def find_next_cell(grid, i, j):
    """
    find the next cell that we can solve
    :param grid:
    :param i:
    :param j:
    :return:
    """
```

```python
def is_valid(grid, row, col, num):
    """
    check if can put num in the board
    :param row: row in board
    :param col: col in board
    :param num: num value
    :return: true if there is no same number in row and col and the square
of num
    """
```

```python
def solve_sudoku(grid, i=0, j=0):
    """
    solve the sudoku grid
    :param grid:
    :param i:
    :param j:
    :return:
    """
```

השימוש בכלי kivy

בעזרת kivy יצרנו את האפליקציה והממשק הכוללים:

- ✓ יצירת המסכים והמעברים ביניהם
- ✓ הוספת וידג'טים סטנדרטים כמו כפתורים, תיבות טקסט, לייבלים, גריד ותמונות
- ✓ הוספת וידג'טים מיוחדים כמו popup , anchor layout , file chooser , toggle button
- ✓ עיצוב הממשק באמצעות רקעים וצבעים
- ✓ שימוש ב object property על מנת לסנכרן בין פעולות למצב האובייקטים

נציג דוגמאות מהקוד בהן נעשה שימוש בכל אחד מהמאפיינים הנ"ל:

- יצירת המסכים והמעברים ביניהם

יצירת מסכים:

בקובץ ה py נגדיר את המשתנים. לצורך הנוחות בדוג' שנביא הגדרנו גם את האלמנטים שבמסך בקובץ ה py כיוון שהמסך כולל בין היתר 81 תאי טקסט, לא נרצה להגדיר כל אחד באופן פרטי בקובץ ה kv אלא להגדיר אותם באמצעות לולאות.

```
class GameWindow(Screen, Widget):
    """
    The Game screen
    Purpose: play from the image that we get from the selection screen
             and show the solution or end the game
    """

    def __init__(self, **kwargs):
        super(GameWindow, self).__init__(**kwargs)
        self.grid = GridLayout()
        self.grid.cols = 1
        self.grid.rows = 2
        self.grid.row_default_height = 582
        self.grid.row_force_default = True
        self.grid.col_default_width = 582
        self.grid.col_force_default = True

        self.big_squares = GridLayout()
        self.big_squares.cols = 3
        self.big_squares.rows = 3
        self.big_squares.spacing = 6
        self.big_squares.padding = 2.5
        self.big_squares.row_default_height = 194
        self.big_squares.row_force_default = True
        self.big_squares.col_default_width = 194
        self.big_squares.col_force_default = True

        for i in range(9):
            self.small_squares = GridLayout()
            self.small_squares.cols = 3
            self.small_squares.rows = 3
            self.small_squares.spacing = 1
            self.small_squares.padding = 1
            self.small_squares.row_default_height = 64.6666667
            self.small_squares.row_force_default = True
            self.small_squares.col_default_width = 64.6666667
            self.small_squares.col_force_default = True

            for j in range(9):
```

```
                row, col = calc_location(i, j)
                cell = Cell(row, col)
                self.small_squares.add_widget(cell)

            self.big_squares.add_widget(self.small_squares)

        self.grid.add_widget(self.big_squares)
        btns_grid = SolveBtnGrid()
        self.grid.add_widget(btns_grid)
        self.add_widget(self.grid)
```

בקובץ הkv נגדיר את העיצוב והאלמנטים שבמסך, כהמשך לדוג' הנתונה האלמנטים כבר מוגדרים בקובץ ה py לכן נותר להגדיר את העיצוב:

```
<GameWindow>:
    id: game
    name: "game"

    canvas.before:
        Color:
            rgba:(0.19,0.19,0.19,1)
        Rectangle:
            pos: self.pos
            size: self.size
```

מעבר בין מסכים:

לשם המעברים מגדירים את ה screen manager

בקובץ py: (לדוג' עבור המחלקה window manager)

```
class WindowManager(ScreenManager):
    pass
```

בקובץ kv נכניס את המסכים:

```
WindowManager:
    MainWindow:
    SelectionWindow:
```

כדי לעבור ממסך ה main למסך ה selection נוכל להגדיר בקובץ ה kv תחת הגדרת הכפתור:

```
on_release:
    app.root.current="selection"
```

- הוספת וידג'טים סטנדרטים כמו כפתורים, תיבות טקסט, ליבלים, גריד ותמונות

כפתור:

בקובץ ה- kv ניתן להגדיר כך לדוג':

```
Button:
    text:"End\ngame"
```

בקובץ ה- py ניתן להגדיר כך לדוג':

```
Button(text="close", size_hint=(None, None), width=200, height=50,
pos_hint={'x': 0, 'y': 0}))
```

תיבת טקסט:

בקובץ ה py לדוג' הגדרנו את המחלקה היורשת מתיבת טקסט:

```
class Cell(TextInput):
…
```

ובקובץ ה kv הגדרנו את העיצוב:

```
<Cell>:
    num: num
    id: num
    halign: "center"
    valign: "middle"
    cursor_color: [0, 0, 0, 0]
    font_size: 0.6 * self.width
    multiline: False
    on_text:
        root.update_num()
```

לייבל:

בקובץ ה kv ניתן להגדיר כך:

```
Label:
    text: "In solve mode any field you\ntouch explore the right number"
    valign: "middle"
    halign: "left"
    font_size: '17sp'
    bold: True
    color:(0.7,0.7,0.7,1)
```

גריד:

בקובץ ה kv לדוג':

```
GridLayout:
    cols: 1
    spacing: 5
    row_default_height:600
    row_force_default: True
```

תמונה:

בקובץ ה kv לדוג':

```
Image:
    source: "images/main_background.jpg"
```

- הוספת וידג'טים מיוחדים כמו toggle button , file chooser , anchor layout , popup

Toggle button :

מאפשר בחירה של xor בין הכפתורים

```
ToggleButton:
    text:"Enter\nsolve\nmode"
    group: "solve_mode"
    halign: "center"
    valign: "middle"
    color:(1,1,1,1)
    width: 70
    size_hint_x: None
    on_press:
        root.enter_solve_mode()
ToggleButton:
    text:"Exit\nsolve\nmode"
    group: "solve_mode"
    halign: "center"
    valign: "middle"
    color:(1,1,1,1)
    width: 70
    size_hint_x: None
    on_press:
        root.exit_solve_mode()
```

File chooser :

מאפשר בחירה של קובץ מהאחסון המקומי

```
FileChooserListView:
    id: filechooser
    on_selection: root.selected(filechooser.selection)
```

```
def selected(self, filename):
    """
    get the image selected
    :param filename: image file
    :return:
    """
    try:
        image_source = filename[0]
```

anchor layout :

מיישר את האלמנטים שבו לגבול (למעלה, למטה, לשמאל, לימין) או למרכז.

```
AnchorLayout:
    anchor_x: 'right'
    Button:
        width: 70
        text:"End\ngame"
        halign: "center"
        valign: "middle"
        color:(1,1,1,1)
        size_hint_x: None
        on_release:
            app.get_running_app().stop()
```

Popup :

הודעה שקופצת דרך חלון על גבי המסך הנתון

```
def show_victory_popup():
    global showed
    showed = True
    layout = FloatLayout()
    msg = "WOWWW !!!\nYou are champion!\nYou have successfully\ncompleted
the game"
    popup_label = Label(text=msg, size_hint=[0.6, 0.2], pos_hint={"x": 0.2,
"top": 0.7}, valign="middle",
                        halign='center', color=[255, 255, 255, 1],
font_size='16sp')
    close_button = Button(text="Close", size_hint=[0.6, 0.2],
pos_hint={"x": 0.2, "y": 0.1})
    layout.add_widget(popup_label)
    layout.add_widget(close_button)

    popup = Popup(title="Congratulations!", title_color=[0, 0, 1, 1],
content=layout, size_hint=[.35, .45],
                  auto_dismiss=False,
                  background='images/popup_win.jpg')
    popup.open()

    close_button.bind(on_press=popup.dismiss)
```

- עיצוב הממשק באמצעות רקעים וצבעים

רקעים :

```
GridLayout:
    cols: 1
    spacing: 5
    row_default_height:600
    row_force_default: True

    Image:
        source: "images/main_background.jpg"
```

צבעים :

```
canvas.before:
    Color:
        rgba:(0.19,0.19,0.19,1)
    Rectangle:
        pos: self.pos
        size: self.size
```

- שימוש ב object property על מנת לסנכרן בין פעולות למצב האובייקטים

לדוג': מגדירים את txt להיות object property ונראה בהמשך שזה מייצג label ומעדכנים את ערכו בקריאה למתודה בהתאם

```
class SolveBtnGrid(GridLayout):
    txt = ObjectProperty(None)

    def enter_solve_mode(self):
        global discovered
        global solution_board

        solve_sudoku(solution_board)
        discovered = True
        self.txt.text = "When you exit from solve mode\nyou continue to
play as usual"

    def exit_solve_mode(self):
        global discovered

        discovered = False
        self.txt.text = "In solve mode any field you\ntouch explore the
right number"
```

והקריאה למתודה נעשית בעת לחיצה על אחד הכפתורים בהתאם

```
<SolveBtnGrid>:
    txt: txt

    …

    AnchorLayout:
        anchor_x: 'left'
        GridLayout:
            cols: 2
            anchor_x: 'left'
            ToggleButton:
                text:"Enter\nsolve\nmode"
```

```
                group: "solve_mode"
                …
                on_press:
                    root.enter_solve_mode()
            ToggleButton:
                text:"Exit\nsolve\nmode"
                group: "solve_mode"
                …
                on_press:
                    root.exit_solve_mode()

    Label:
        id: txt
        text: "In solve mode any field you\ntouch explore the right number"
        …
```

השימוש בכלי opencv

עיבוד התמונה (שימוש בכלי OpenCV וב-numpy array):

בעזרת openCV מעבדים את התמונה המתקבלת מהמשתמש:

- ✓ עיבוד התמונה המקורית על מנת לקבל את לוח הסודוקו מתוכה.
- ✓ עיבוד כל תא מתוך לוח הסודוקו על מנת לוודא שקיימת סיפרה בתא.
- ✓ קבלת הגבולות של הספרה בתוך התא, כדי שנוכל ליצור מערך numpy שמכיל את הפיקסלים של הספרה.

נציג דוגמאות מהקוד בהן נעשה שימוש בכל אחד מהמאפיינים הנ"ל:

עיבוד התמונה המקורית ע"מ לקבל את לוח הסודוקו:

לצורך כך השתמשנו בפונקציות הבאות מתוך OpenCV:

```
gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
```

פונקציה זו משנה את גוון התמונה לאפור כדי שנוכל בסופו של דבר להפוך את התמונה לבינארית(-שחור-לבן)

```
cv.adaptiveThreshold(gray, 255, cv.ADAPTIVE_THRESH_GAUSSIAN_C,
cv.THRESH_BINARY_INV, 11, 9)
```

בפונקציה זו מחשבים את ערך הסף עבורו ייקבע האם הפיקסל יהיה שחור או לבן בעזרת 11 שכנים של כל פיקסל.

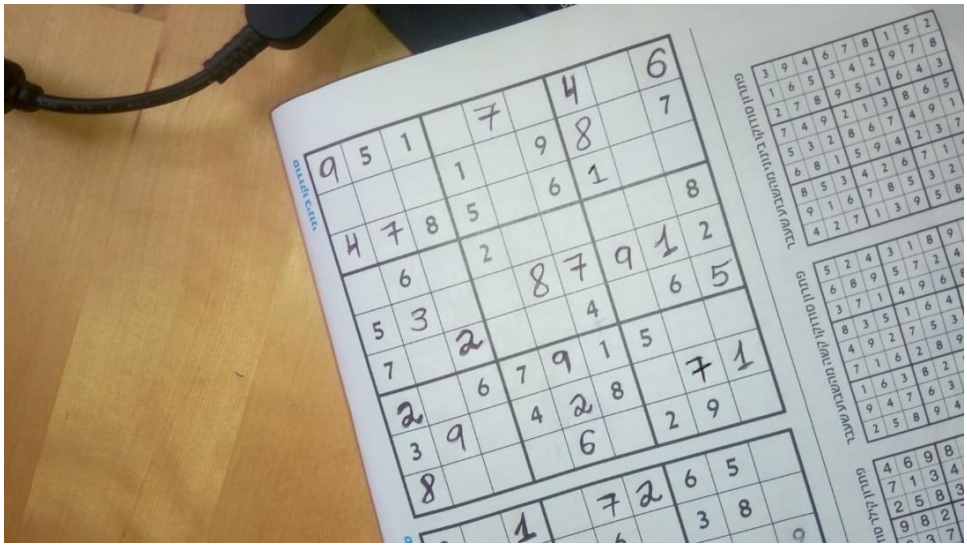
```
cv.findContours(adaptive_thresh, cv.RETR_EXTERNAL, cv.CHAIN_APPROX_SIMPLE)
```

פונקציה זו מחזירה את כל הדמויות שבתמונה הבינארית שחושבה כפי שתיארתי, כלומר את קבוצות הפיקסלים שמגדירות צורה/ דמות כלשהי בתמונה.

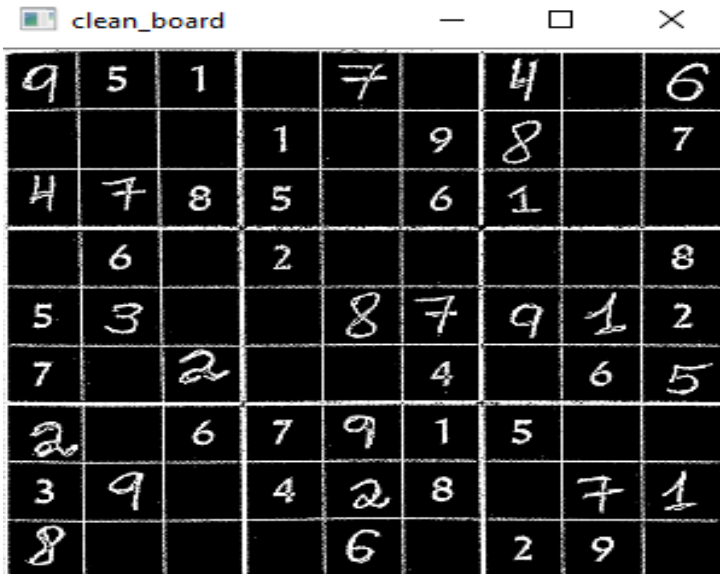
```
cv.arcLength(sorted_contours[i], True)
cv.approxPolyDP(sorted_contours[i], epsilon, True)
```

פונקציות אלו מחזירות את היקף הצורה/דמות ואת הצורה המוכרת שהכי דומה לה (מידת התאימות תלויה במשתנה-epsilon) שמכילה פחות קודקודים. פונקציות אלו סייעו לנו לזהות צורה מתוך כלל הצורות שמצאנו שהיא בעלת 4 קודקודים, כלומר מלבן.

לדוגמא- עבור התמונה הבאה-



נקבל את התמונה המעובדת הבאה-



לאחר מכן, חילקנו את אורך ורוחב הלוח שקיבלנו ב-9, עברנו על כל חלקי התמונה- התאים ושלחנו אותם לעיבוד בפונקציה 'get_num' .

עיבוד כל תא כלל את הפונקציות הבאות-

```
clear_border(thresh, buffer_size=25)
```

בפונקציה זו השתמשנו על מנת לנקות את גבולות התא מהפיקסלים של הקוים המפרידים בין התאים.

```
contours_digit, hierarchies_digit = cv.findContours(thresh1.copy(),
cv.RETR_EXTERNAL, cv.CHAIN_APPROX_SIMPLE)
sorted(contours_digit, key=lambda x: cv.contourArea(x), reverse=True)
```

בפעולות אלו מוצאים את כל הצורות בתא וממיינים אותן לפי שטח הצורות.

```
blank = np.zeros(thresh1.shape, dtype='uint8')

cv.drawContours(blank, [sorted_contours_digits[0]], -1, 255, -1)
not_zeros = cv.countNonZero(blank) /
float(thresh1.shape[0]*thresh1.shape[1])
```

בשורות אלו חישבנו את החלק היחסי של הצורה בעלת השטח הכי גדול (אמורה להיות הספרה, אם קיימת) מכלל הפיקסלים בתא ואם החלק היחסי עולה על 0.04 משמע שיש שם ספרה ואחרת יש רק "רעשים" בתא.

```
x_corner, y_corner, w, h = cv.boundingRect(sorted_contours_digits[0])
```

בפונקציה זו השתמשנו על מנת לקבל את הגבולות של הצורה בעלת השטח הדומיננטי בתא.

משלב זה ואילך אתאר את השימוש בייצוג של numpy array ע"מ לתאר את הספרה.

```
x_corner, y_corner, w, h = cv.boundingRect(sorted_contours_digits[0])
digit = thresh[y_corner:y_corner + h, x_corner:x_corner + w]
left, right, upper, lower = find_bounderies(digit)
width = right - left + 1
height = lower - upper + 1
```

בשורות אלו, מצאתי את גבולות הספרה בתא בעזרת הפונקציה 'find_contour' בה עוברים על הפיקסלים שמרכיבים את הספרה עד שמגיעים לקצה הספרה וקובעים באמצעותו את הגבול.

```
square_number = create_square_around_digit(digit, height, width, upper,
left)
padded_number = paddig_with_black_pixels(square_number)
```

בפונקציה 'create_square_around_digit' יוצרים ריבוע סביב פיקסלי התמונה – כלומר מרפדים את האורך או הרוחב בהתאם לגודלם בפיקסלים שחורים.

בפונקציה 'padding_with_black_pixels' מרפדים את הריבוע שקיבלנו בשלב הקודם בלפחות 4 פיקסלים מכל כיוון כך שהתמונה תתאים למודל רשת הנוירונים שבאמצעותו חוזים את ערך הספרה בתא.

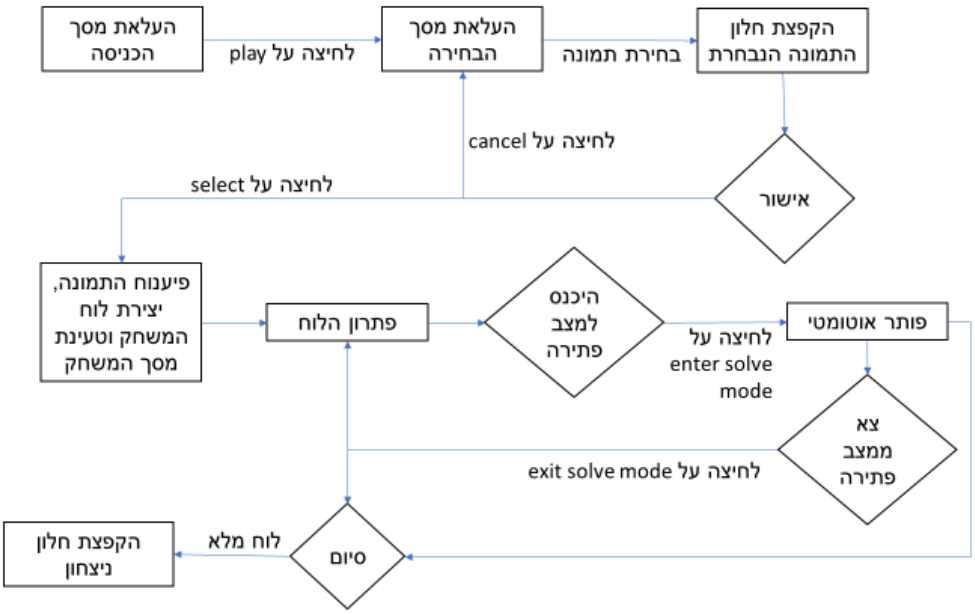
אחר הצעדים שתוארו לעיל, חוזרים לפונקציה 'process_cells' בה מבצעים חיזוי לתא המעובד כפי שניתן לראות בשורה הבאה-

```
prediction = model.predict(number).argmax(axis=1)[0]
```

תוצאות

תוצאות השימוש בכלי kivy

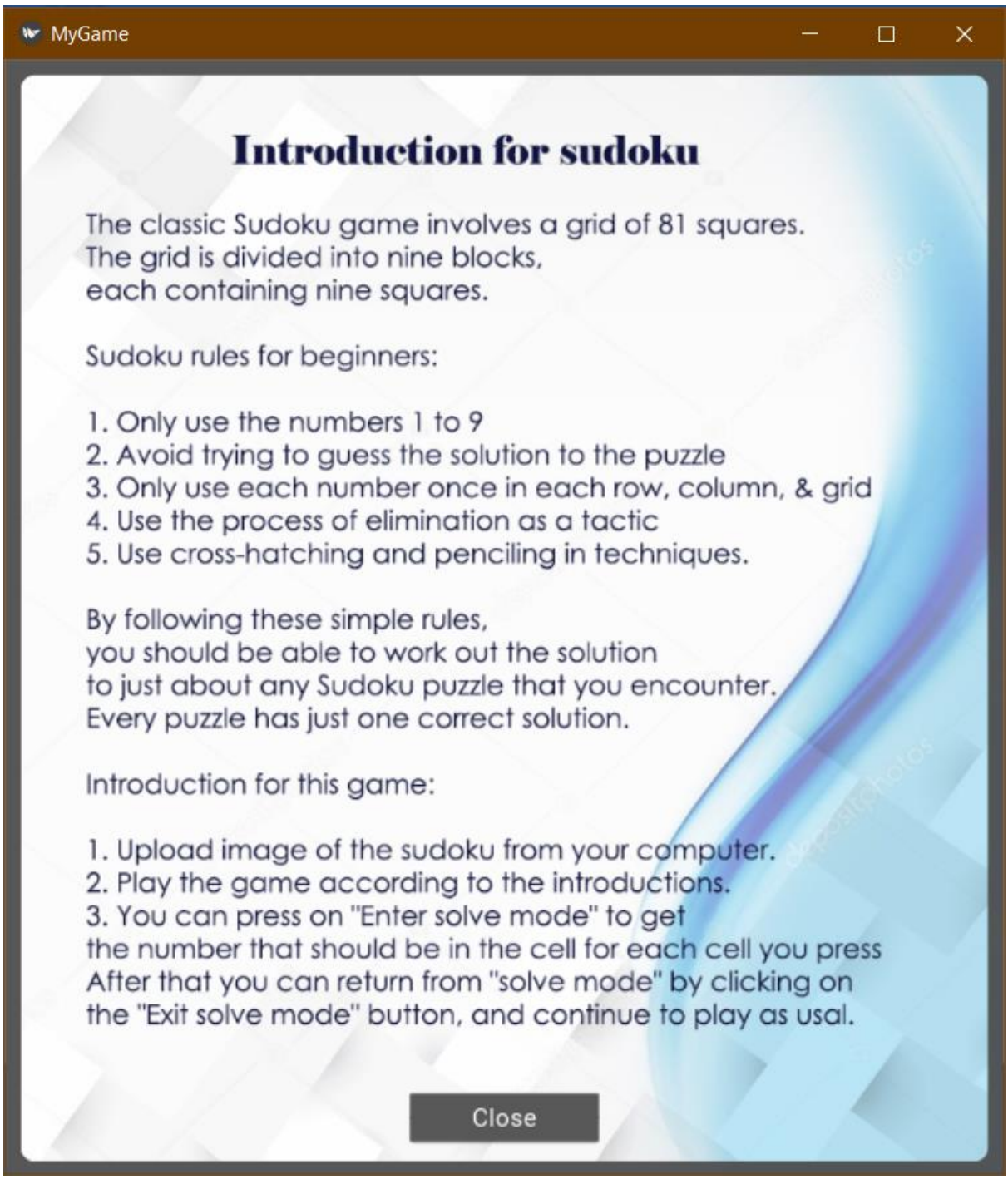
מבנה התהליך המתבצע עם הרצת התוכנית



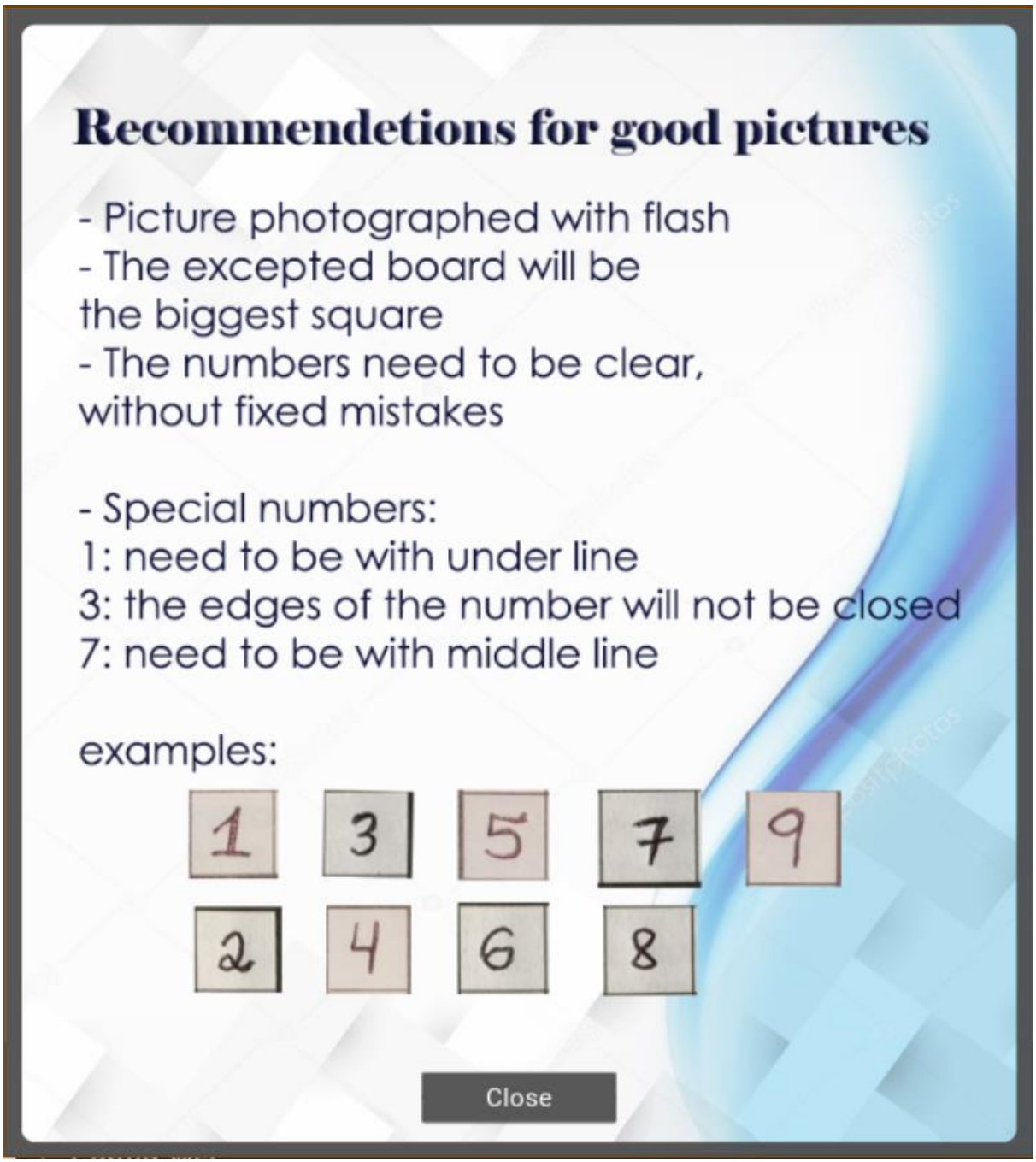
מסך ראשי:


MyGame
Digital sudoku solver
Camera
Upload your image of sudoku
and continue to play here!
Recommends
PLAY
About

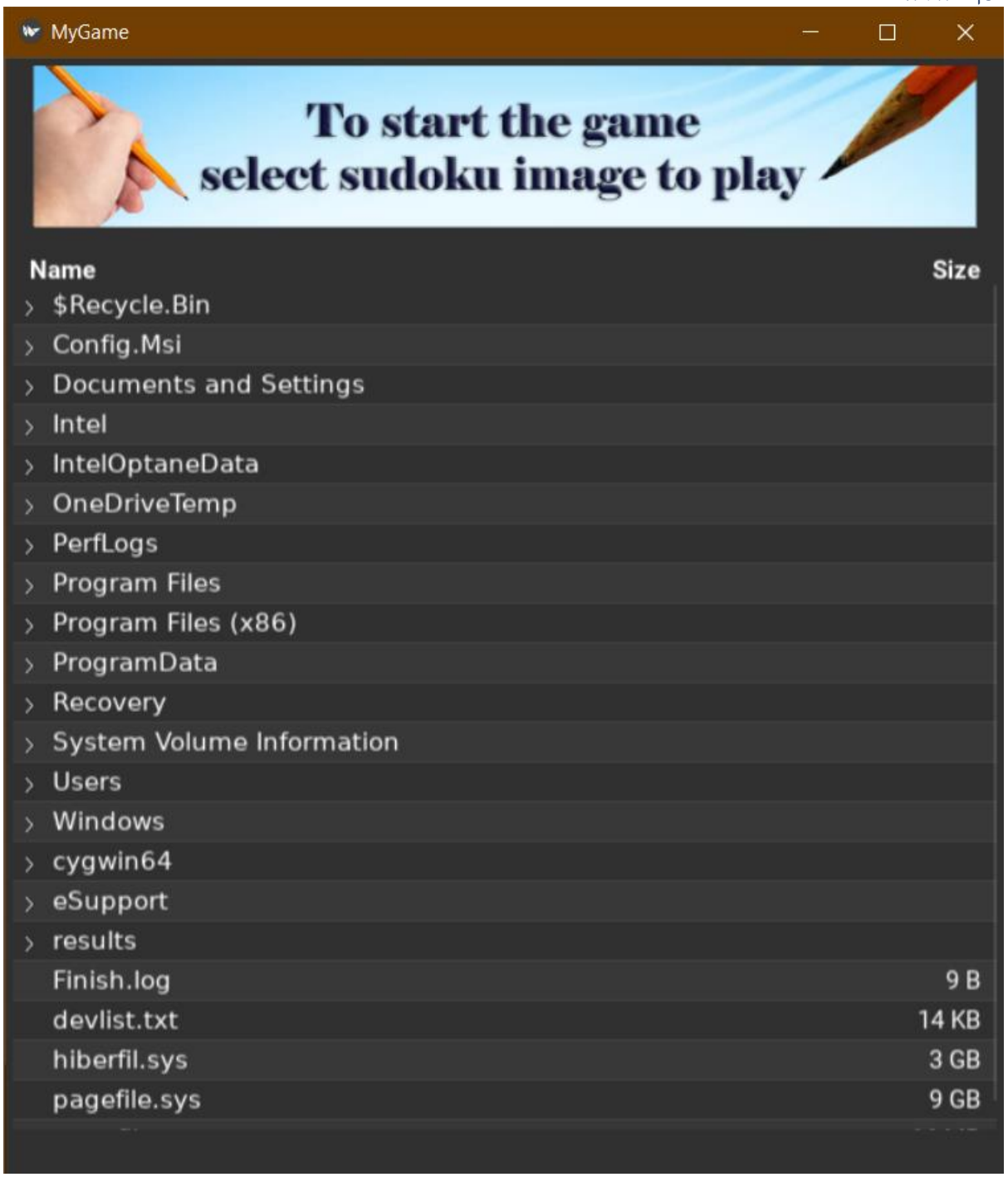
: About


MyGame
Introduction for sudoku
The classic Sudoku game involves a grid of 81 squares.
The grid is divided into nine blocks,
each containing nine squares.
Sudoku rules for beginners:
1. Only use the numbers 1 to 9
2. Avoid trying to guess the solution to the puzzle
3. Only use each number once in each row, column, & grid
4. Use the process of elimination as a tactic
5. Use cross-hatching and penciling in techniques.
By following these simple rules,
you should be able to work out the solution
to just about any Sudoku puzzle that you encounter.
Every puzzle has just one correct solution.
Introduction for this game:
1. Upload image of the sudoku from your computer.
2. Play the game according to the introductions.
3. You can press on "Enter solve mode" to get
the number that should be in the cell for each cell you press
After that you can return from "solve mode" by clicking on
the "Exit solve mode" button, and continue to play as usal.
Close

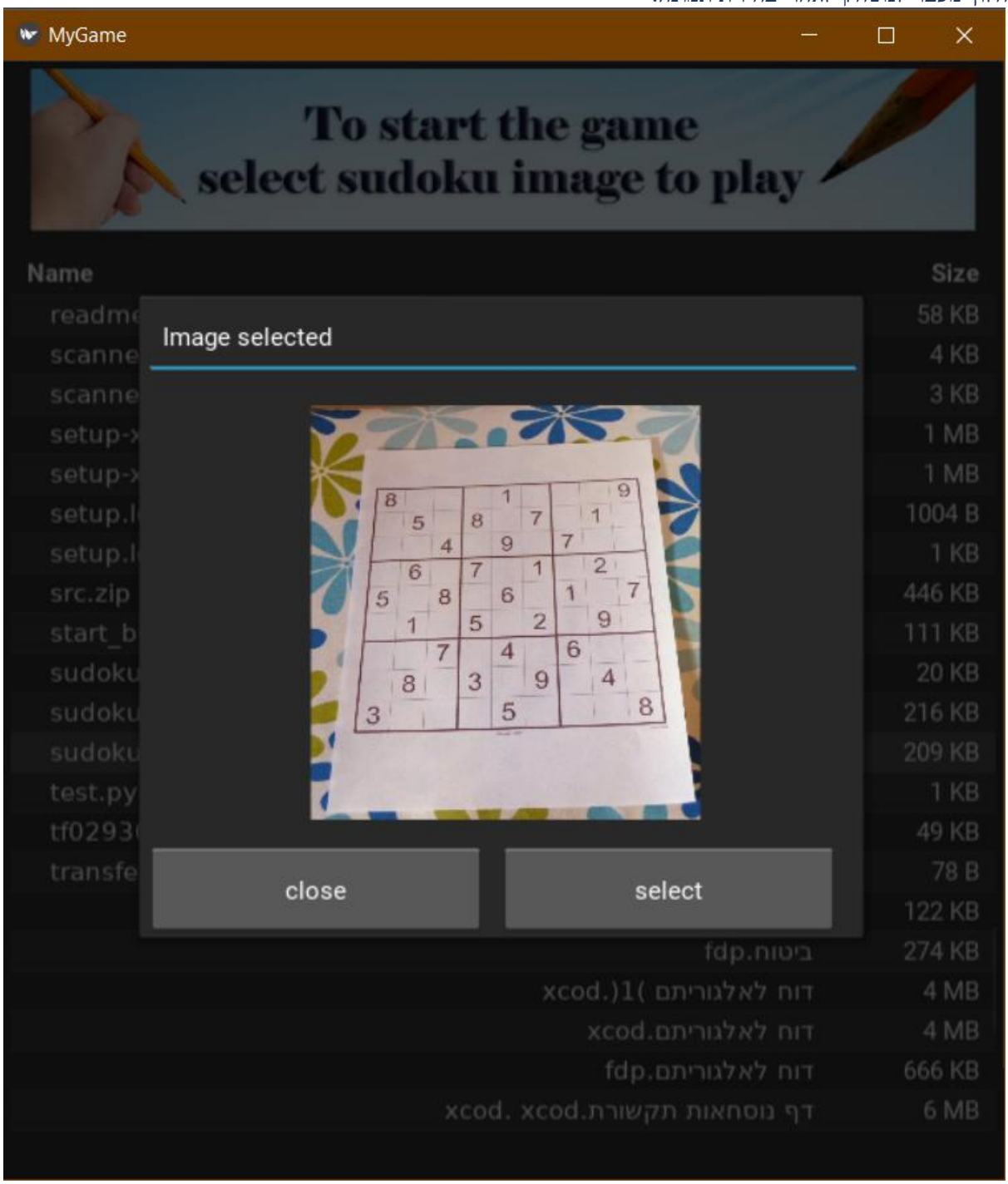
: Recommends


Recommendetions for good pictures
- Picture photographed with flash
- The excepted board will be
the biggest square
- The numbers need to be clear,
without fixed mistakes
- Special numbers:
1: need to be with under line
3: the edges of the number will not be closed
7: need to be with middle line
examples:
1 3 5 7 9
2 4 6 8
Close

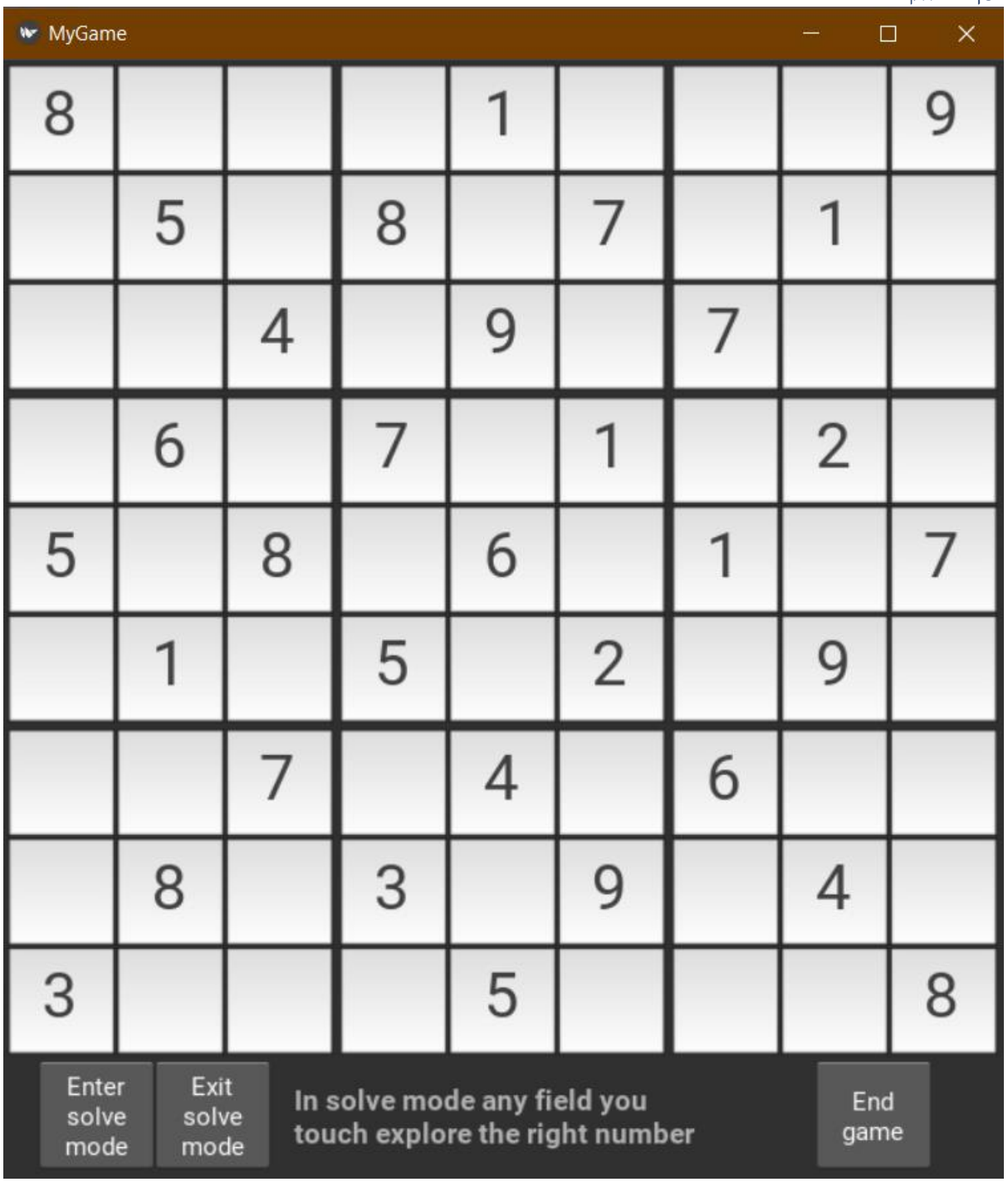
מסך בחירה:


MyGame
To start the game
select sudoku image to play
Name
Size
$Recycle.Bin
Config.Msi
Documents and Settings
Intel
IntelOptaneData
OneDriveTemp
PerfLogs
Program Files
Program Files (x86)
ProgramData
Recovery
System Volume Information
Users
Windows
cygwin64
eSupport
results
Finish.log 9 B
devlist.txt 14 KB
hiberfil.sys 3 GB
pagefile.sys 9 GB

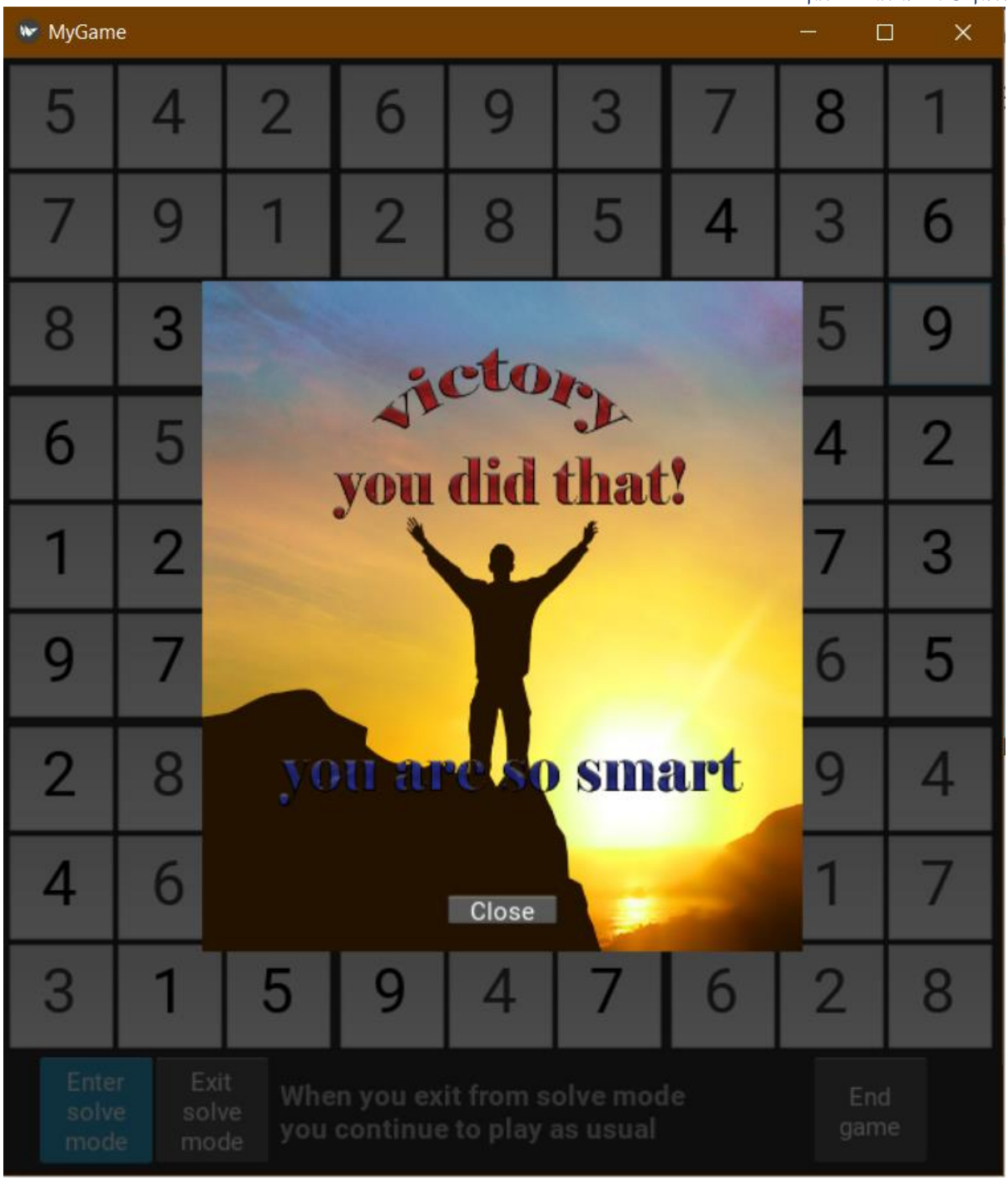
חלון מעבר למשחק לאחר בחירת תמונה:



מסך משחק:

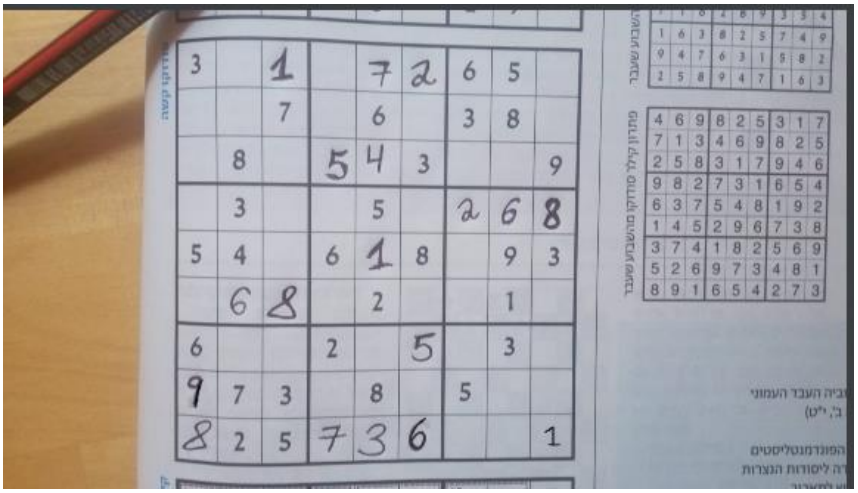

MyGame
8 1 9
5 8 7 1
4 9 7
6 7 1 2
5 8 6 1 7
1 5 2 9
7 4 6
8 3 9 4
3 5 8
Enter solve mode
Exit solve mode
In solve mode any field you touch explore the right number
End game

חלון סיום לאחר ניצחון:

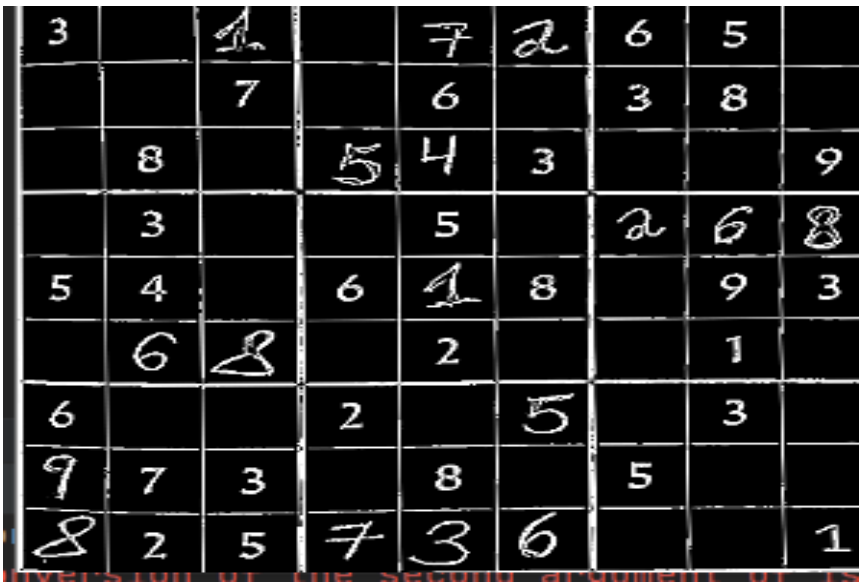


תוצאות השימוש בכלים לעיבוד תמונה

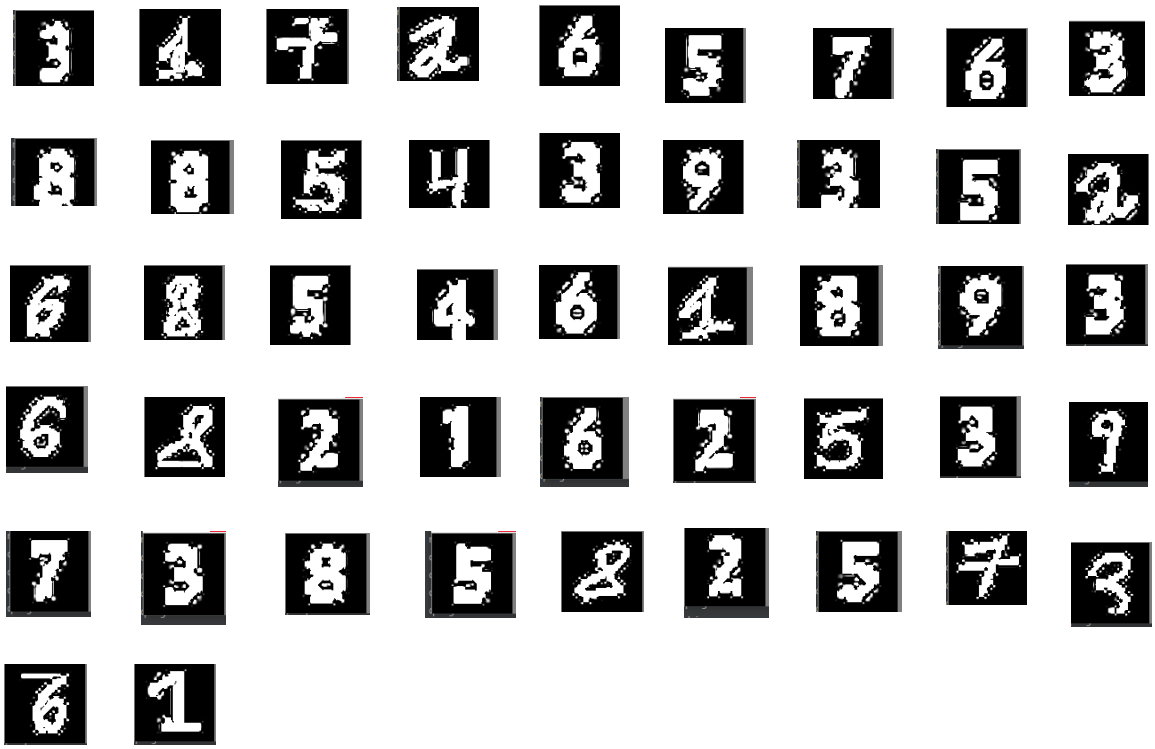
לשם המחשה, עבור התמונה הבאה



לאחר עיבוד התמונה הראשוני שמטרתו להתמקד בלוח עצמו נקבל את התוצאה הבאה



התאים המועבדים שישלחו למודל רשת הנוירונים יראו כבתמונות שלהלן-



בנייה ואימון רשת נוירונים לזיהוי ספרות

ניתוח המידע

ראשית ננתח את הנתונים שלנו

```
[INFO] get data from MNIST...
X_train shape is(28 ,28 ,60000) :
y_train shape is(,60000) :

X_test shape is(28 ,28 ,10000) :
y_test shape is(,10000) :
```

נציג מעט תמונות ממאגר האימון שלנו

לאחר היכרות עם הנתונים ניגש כעת לבניית המודל

היה ביכולתנו לבנות מודל יותר מורכב, או להשתמש במודל קיים כמו resnet או vgg ולבצע transfer learning אך רצינו לבנות מודל פשוט ככל הניתן, מהסיבה הפשוטה,

כיוון שמדובר בבעיית קלסיפיקציה של תמונות של ספרות בהן אין הרבה רעשים, צבעים, ונושאים בעלי מכנה משותף שנרצה להתעלם מהן בהכללה. אזי פעולת החיזוי תהיה פשוטה יותר.

בניית המודל

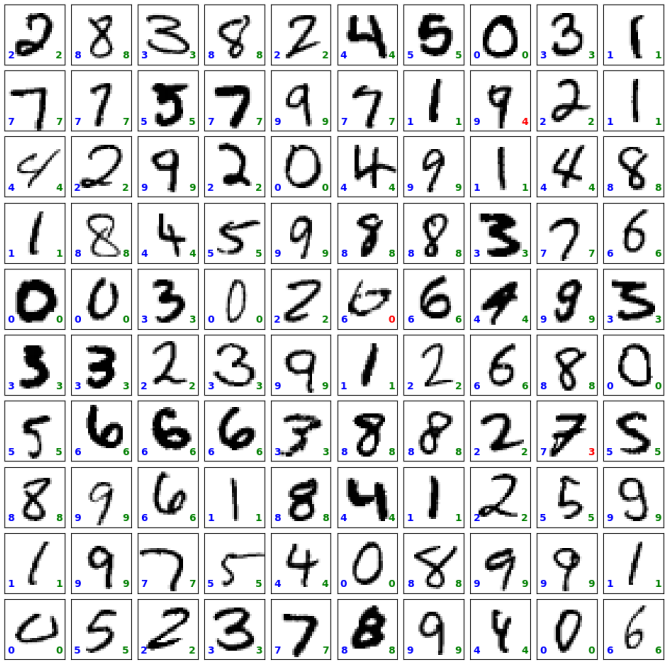
```
[INFO] building model...
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param#
=================================================================
conv2d (Conv2D)              (None, 26, 26, 32)        320
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 24, 24, 32)        9248
_________________________________________________________________
dropout (Dropout)            (None, 24, 24, 32)        0
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 12, 12, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 10, 10, 16)        4624
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 8, 16)          2320
_________________________________________________________________
dropout_1 (Dropout)          (None, 8, 8, 16)          0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 4, 4, 16)          0
_________________________________________________________________
flatten (Flatten)            (None, 256)               0
_________________________________________________________________
dense (Dense)                (None, 10)                2570
=================================================================
Total params: 19,082
Trainable params: 19,082
Non-trainable params: 0
_________________________________________________________________
[INFO] compiling model...
```

```
[INFO] training CNN...

Train on 48000 samples, validate on 12000 samples

Epoch 1/10

2020-12-21 :00:23:04.127642 I
tensorflow/core/profiler/lib/profiler_session.cc:225] Profiler
session started.

48000/4800054 - [==============================] s 1ms/sample -
loss: 0.2337 - acc: 0.9265 - val_loss: 0.0856 - val_acc: 0.9783

Epoch 2/10

48000/4800053 - [==============================] s 1ms/sample -
loss: 0.0759 - acc: 0.9770 - val_loss: 0.0634 - val_acc: 0.9813

Epoch 3/10

48000/4800053 - [==============================] s 1ms/sample -
loss: 0.0543 - acc: 0.9832 - val_loss: 0.0618 - val_acc: 0.9824

Epoch 4/10

48000/4800053 - [==============================] s 1ms/sample -
loss: 0.0449 - acc: 0.9859 - val_loss: 0.0482 - val_acc: 0.9865

Epoch 5/10

48000/4800056 - [==============================] s 1ms/sample -
loss: 0.0374 - acc: 0.9881 - val_loss: 0.0375 - val_acc: 0.9889

Epoch 6/10

48000/4800058 - [==============================] s 1ms/sample -
loss: 0.0335 - acc: 0.9885 - val_loss: 0.0400 - val_acc: 0.9887

Epoch 7/10

48000/4800053 - [==============================] s 1ms/sample -
loss: 0.0290 - acc: 0.9909 - val_loss: 0.0387 - val_acc: 0.9893

Epoch 8/10

48000/4800053 - [==============================] s 1ms/sample -
loss: 0.0267 - acc: 0.9916 - val_loss: 0.0380 - val_acc: 0.9887

Epoch 9/10

48000/4800052 - [==============================] s 1ms/sample -
loss: 0.0256 - acc: 0.9915 - val_loss: 0.0311 - val_acc: 0.9911

Epoch 10/10

48000/4800053 - [==============================] s 1ms/sample -
loss: 0.0222 - acc: 0.9933 - val_loss: 0.0364 - val_acc: 0.9887

[INFO] evaluating CNN...
```

```
model accuracy on test set is: 99.17%
```

[INFO] serializing digit model...

נציג מעט תמונות מחיזוי נתוני הטסטים



מסקנות

כלי kivy

יתרונות:

- מצאנו בכלי הפיתוח ייצוג מאוד רחב של אלמנטים ויכולות עיצוב מאוד מעניינות ועשירות.
- גילינו אלמנטים חדשים, כמו toggle button , file chooser וכדו'.
- גילינו דרכים מעניינות לעצב את האפליקציה, ביניהם עיצוב רקעים והכנסתם כתמונה או כרקע לליבל, שינוי צבעי הרקע, מעבר בין מסכים וכו'. זה הפתיע אותנו לטובה. לא ציפינו בהתחלה שתצא לנו אפליקציה מעוצבת כל כך, לא הערכנו ש kivy מסוגלת להגיע לתוצאות מאוד מרשימות.
- מצאנו דרך יעילה להגדיר את כל 81 התאים. במקום להגדיר כל אחד בקובץ ה-kv הגדרנו דרך לולאה מקוננת בקובץ ה-py. הכלי מאפשר גמישות בהגדרת האלמנטים ע"י קובץ kv ו-py.

חסרונות:

- עסקנו בחלוקה לגדלים שונים של grids. מבחינה זו ישנה מעט ביקורת על הכלי, החלוקה נעשית בצורה לא טבעית, וצריך לבצע הרבה קונספירציות עד שמגיעים לגדלים והמיקומים הרצויים.
- רצינו להעביר מידע למסך כך שבאתחול המסך הוא ישתמש במידע הזה. לכן רצינו שהאתחול יקרה במעבר למסך הנ"ל. הדבר לא יתאפשר כיוון שיצירת המסך מתבצעת מראש עם הגדרת ה-screenManager. היה מאוד טוב אם היה מאפיין בוליאני לכלי המסמל האם לאתחל את המסך מראש או בעת מעבר אליו.
הדרך בה התגברנו על הבעיה, היה לא להכניס את המסך מראש תחת ה-screenManager , אלא ליצור אותו בטרם המעבר אליו, עם הנתונים הנצרכים, ורק לאחר מכן להכניס אותו תחת ה-screenManager ולעבור אליו.

הספרייה openCV

יתרונות:

- באמצעות הספרייה הזו ניתן בקלות לזהות אובייקטים בצורות רצויות בתמונה.
- באמצעותה ניתן לודא שאכן קיים בחלק מהתמונה אובייקט (בפרויקט שלנו -וידוי המצאות ספרה בתא)
- באמצעותה ניתן בקלות לנקות את קצות התמונה מ-"רעשים" ולהתרכז רק בתוכן עצמו וכן ניתן למצוא גבולות של צורות בתוך התמונה, בפרויקט זה סייע בהכנת התמונה לקראת חיזוי ערך הספרה שבתמונה.

חסרונות:

- בפרויקט שלנו עסקנו בתמונות שהמשתמש מספק, לכן המערכת צריכה לתמוך בתמונות מסוגים שונים שצולמו תחת תנאים שונים, אבל כשמשתמשים בספרייה openCV נדרשות לעיתים התאמות של פרמטרים מסוימים- למשל איזו שיטה לספק לפונקציה adaptive_threshold- האם 'ADAPTIVE_THRESH_MEAN_C' או 'ADAPTIVE_THRESH_GAUSSIAN_C' וכן ע"פ כמה שכנים לחשב את ערך הסף ומה הערך שנוריד מהתוצאה הסופית (הפרמטר השישי).
- בתמונות שונות גם ערך ה-epsilon (שבאמצעותו בודקים איזו צורה קרובה לדמות שבתמונה בפונקציה approxPolyDP) עלול להיות שונה.
- יכולות להיות תמונות שבהן קצוות התאים – כלומר הקוים שמפרידים בין התאים, יהיו בעובי שונה ועל כן ערך ה-buffer_size שמספקים לפונקציה clear_border עלול להיות שונה.

- בחלק שבו מעוניינים לזודא שאכן קיימת ספרה בתא מבצעים בדיקה על החלק היחסי של המורה מתוך כלל התא ולדוגמא בפרויקט שלנו קבענו מתוך ניסוי וטעייה את ערך הסף להיות 0.04 , אבל יכולות להיות תמונות בהן נפספס סיפרה כיוון שהכתב היה דק מאוד.

<u>הגבלות על הקלט:</u>

- התמונות אמורות להילקח בתנאי תאורה טובים(שימוש בפלאש).
- עדיף שהאחדות יהיו בעלות קו תחתון –
  1
- ושבעיות יהיו מהצורה בה יש קו גם באמצע הספרה-
  

כלי keras

<u>יתרונות:</u>

- מאפשר ניתוח מעמיק של המידע ומגיע ליכולות חיזוי גבוהות
- מכיל המון פיצ'רים ופרמטרים המאפשרים לשפר את הפרדיקציות, השתמשנו בחלקם כפי שניתן לראות בקוד, אולם זוהי תורה שלימה שניתן להעמיק בה, ולהגיע מאוד רחוק.
- מודל פשוט יחסית הביא לתוצאות מאוד יפות.

<u>חסרונות:</u>

- אם כל זאת שאנו מאמנים מודל עם accuracy ממש גבוה, בתוצאות זמן האמת, עבור תמונות שונות, ישנה שונות, והוא טועה במספר פעמים, הדרך לשפר אותה היא לבצע אוגמנטציה של הדטא ובכך להתאים את האימון על מגוון יותר רחב של ניסויים אפשריים, פעולות אלו מצריכות משאבים מאוד חזקים של המחשב.