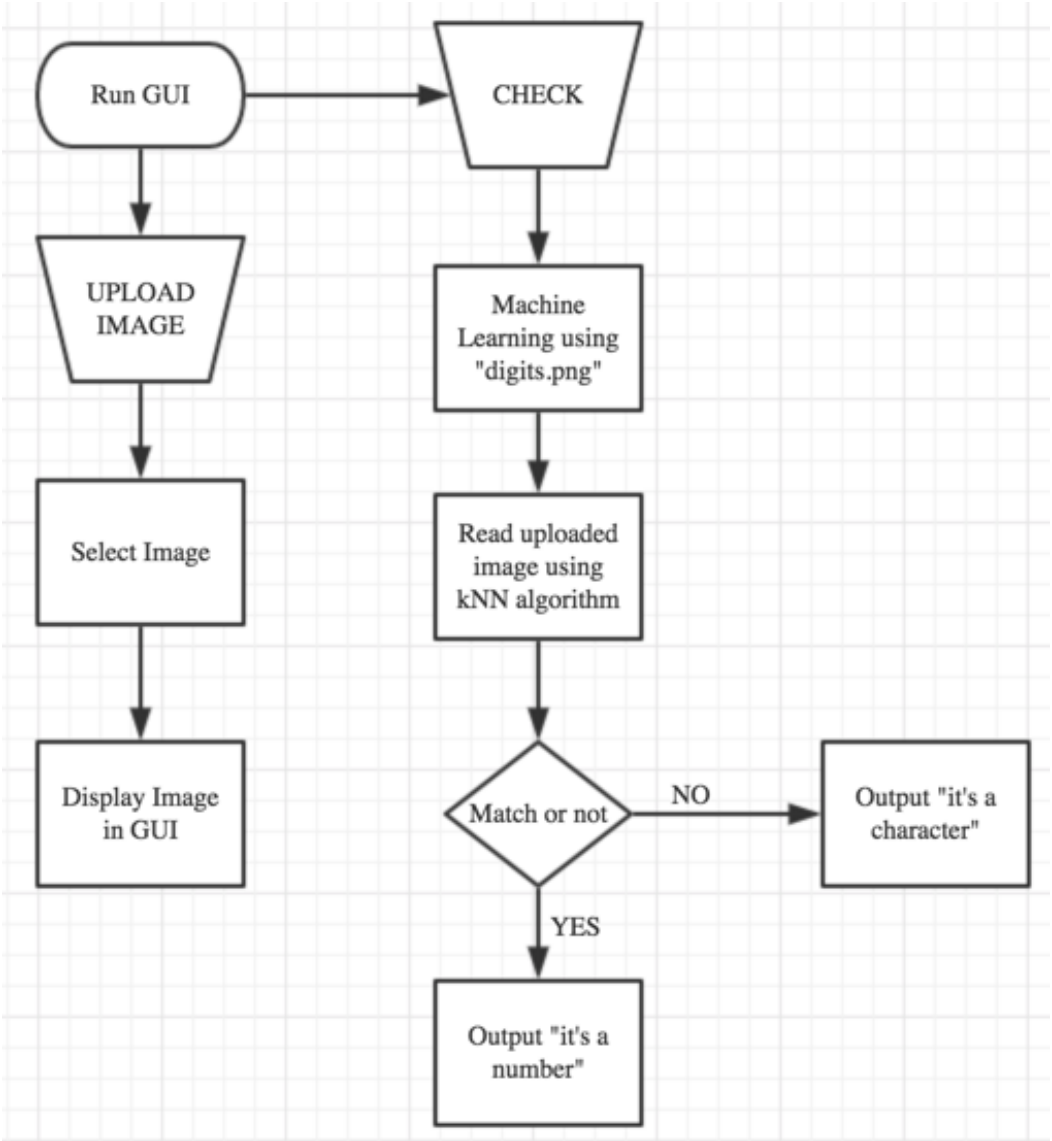
Python Final Project Report
"Number or Character" - Recognition Program
Kaushal Patel, Zhe Zhou, Zhaoyang Xie

Introduction

"Number or Character" – Recognition Program is a GUI that consists of 2 buttons. "UPLOAD IMAGE" button lets the user select an image they want to check/classify. That image is then displayed in the GUI under "Selected Image (shown below):". "CHECK" buttons trigger the machine learning where the computer reads the image "digits.png" to learn what numbers between 0 to 9 should like. "digits.png" consists of 50 rows & 100 columns with 500 variations of each number. After the learning process is complete, program uses the kNN (kth Nearest Neighbor) algorithm to calculate whether the uploaded image is a number or a character (not a number). Then the result is displayed in the GUI under "Number or Character (shown below):".



Python Final Project Report
"Number or Character" - Recognition Program
Kaushal Patel, Zhe Zhou, Zhaoyang Xie

Project Results

The original objective was to use machine learning algorithm(s) to be able to recognize the character/number uploaded by the user & output the results. However, due to time constraints and the complexity of the algorithm we were only able to reach the point where the program can teach itself how to recognize few numbers. Also the characters that look like numbers are unusable.

One of the pitfalls we encountered was when we discovered that kNN algorithm is only able to tell whether it has found the nearest neighbor of the uploaded image; this algorithm, by itself, cannot isolate the character/number drawn on the uploaded image.

To improve the accuracy of this program, it needs more learning data. Thus, increasing the learning data from 5000 to more than 10,000 would significantly increase its accuracy. Once the accuracy has been improved, condition statements and accuracy (%) results can be used to isolate/recognize and output the character/number drawn on the uploaded image.

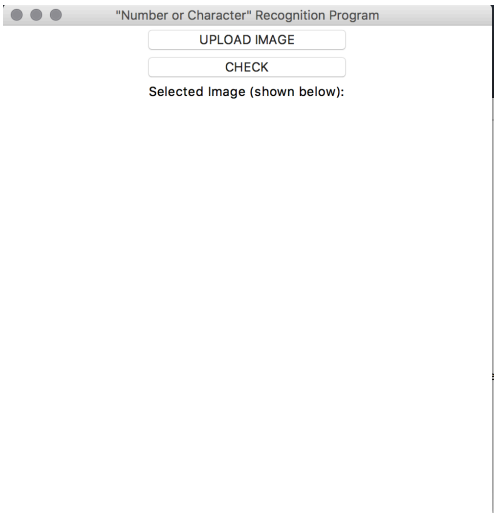


Fig. 1

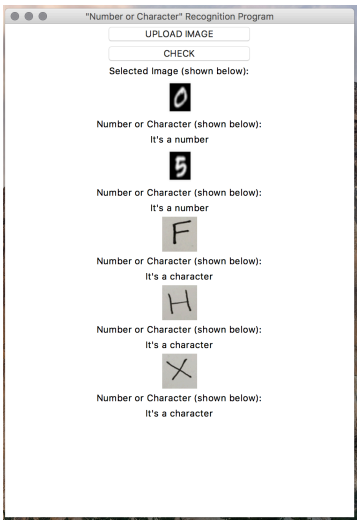


Fig. 2

Figure 1 shows the beginning of the GUI. It consists of 2 buttons "UPLOAD IMAGE" and "CHECK". As shown in Figure 2, after user selects an image to upload, that image is shown below the label "Selected Image (shown below):" and after checking whether the uploaded image is a number or character (using kNN algorithm) the result is displayed below the label "Number or Character (shown below):". Figure 2 shows how the program can recognize the uploaded number-images as numbers and letters-images as characters.

Python Final Project Report
"Number or Character" - Recognition Program
Kaushal Patel, Zhe Zhou, Zhaoyang Xie

Division of Group Work

Zhaoyang Xie – Coded the GUI (version 1) layout.

For this project, I created GUI (version 1) by using the Tkinter. The GUI (version 1) has two functional buttons. The first button is the "Train", which is designed to train the program using kNN functions. The other functional button is "Upload an image"; this button can select both PNG and GIF file types, and then display them out onto the screen.

Zhe Zhou – Found how to use & coded kNN algorithm in python.

I found this KNN algorithm concept and introduced to my teammates. And after the research I have done, I found the OpenCV package is a perfect match for us. It took me a week to successfully install the OpenCV package, and I helped my teammates install that package. I also developed the main frame of the program, and helped my teammates troubleshooting on their part.

Kaushal Patel – Update previous GUI (version 1) to GUI (version 2). Coded the process of using kNN on the uploaded image to recognize whether it's a number or a character.

For this project, I created GUI (version 2) which has "UPLOAD IMAGE" and "CHECK" buttons, "Selected Image (shown below):" and "Number of Character (shown below):" labels. The "UPLOAD IMAGE" button lets the user select an image and displays it in the GUI. "CHECK" button initiates the machine learning, check whether the selected image is a number or a character using kNN algorithm and displays the result in the GUI.

Python Final Project Report
"Number or Character" - Recognition Program
Kaushal Patel, Zhe Zhou, Zhaoyang Xie
Bibliography

Python GUI Tutorial:
https://www.tutorialspoint.com/python/python_gui_programming.htm

Python GUI YouTube Tutorial:
https://www.youtube.com/watch?v=RJB1Ek2Ko_Y&list=PL6gx4Cwl9DGBwibXFtPtflztSNPGuIB_d

Install Brew
http://brew.sh

Install OpenCV3
http://www.pyimagesearch.com/2015/06/29/install-opencv-3-0-and-python-3-4-on-osx/

Install Pillow
Pip3 install pillow

Install tkinter
Python in Mac already comes with tkinter (this is what we used)

Script to run
brew install homebrew/science/opencv3 --HEAD --with-python3 --with-ffmpeg --with-tbb --with-contrib
echo /usr/local/opt/opencv3/lib/python3.5/site-packages >> ~/.virtualenvs/cv3/lib/python3.5/site-packages/opencv3.pth
brew install pyqt5
echo /usr/local/opt/sip/lib/python3.5/site-packages >> ~/.virtualenvs/cv3/lib/python3.5/site-packages/sip.pth
echo /usr/local/opt/pyqt5/lib/python3.5/site-packages >> ~/.virtualenvs/cv3/lib/python3.5/site-packages/pyqt5.pth
pip3 install matplotlib

Python Final Project Report
"Number or Character" - Recognition Program
Kaushal Patel, Zhe Zhou, Zhaoyang Xie

Code Appendix

[NOTE: Code we wrote vs. Code found on the internet (stackoverflow.com, etc.)]

```
from tkinter import *
from tkinter.filedialog import askopenfilename
from tkinter import PhotoImage
from PIL import Image, ImageTk
import numpy as np
import cv2
import matplotlib
matplotlib.use("Qt5Agg")
from matplotlib import pyplot as plt

class Uploader(Tk):
    def __init__(self,*args, **kwargs):
        Tk.__init__(self, *args, **kwargs)
        self.root = Canvas()
        self.root.pack()

        #creating buttons & labels
        uploadButton = Button(self.root, text='UPLOAD IMAGE', command=self.upload_image).pack(fill=X)
        convertButton = Button(self.root, text='CHECK', command=self.check_for_match).pack(fill=X)
        imageLabel = Label(self.root, text='Selected Image (shown below):').pack()

    #uploading an image
    def upload_image(self):
        file_name = askopenfilename(filetypes=[('PNG files', '*.png'), ('JPEG FILES', '*.jpg')])
        image = Image.open(file_name)
        image = image.resize((50, 50), Image.ANTIALIAS)
        photo = ImageTk.PhotoImage(image)
        displayImage = Label(image=photo)
        displayImage.image = photo
        displayImage.pack()

        self.file_name = file_name

    #read & convert the uploaded image
    def check_for_match(self):
        textLabel = Label(text='Number or Character (shown below):').pack()

        #reading the training image
        img = cv2.imread('digits.png')
        gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
        plt.imshow(gray, cmap = 'gray', interpolation = 'bicubic')
        plt.xticks([]), plt.yticks([])  # to hide tick values on X and Y axis

        #split the gray image to 5000 cells into 50 rows and, each 20*20 size
        cells = [np.hsplit(row,100) for row in np.vsplit(gray,50)]

        #Make it into a Numpy array. It size will be (50,100,20,20)
        x = np.array (cells)

        #training AI
        train = x[:,:100].reshape(-1,400).astype(np.float32)
        k = np.arange(10)
        train_labels = np.repeat(k,500)[:,np.newaxis]d

        #input sample
        sample = cv2.imread(self.file_name)
        gray2 = cv2.cvtColor(sample, cv2.COLOR_BGR2GRAY)

        #resize input image to 20x20
        resizedSample = cv2.resize(gray2, (20, 20), interpolation = cv2.INTER_CUBIC)
        plt.imshow(resizedSample, cmap = 'gray', interpolation = 'bicubic')
        plt.xticks([]), plt.yticks([])

        #put image in 1 cell
        sampleCells = [np.hsplit(row, 1) for row in np.vsplit(resizedSample, 1)]
        y = np.array(sampleCells)
        test = y[0][0].reshape(-1, 400).astype(np.float32)

        #knn algorithm
        knn = cv2.ml.KNearest_create()
        knn.train(train,cv2.ml.ROW_SAMPLE,train_labels)
        #test it with test data for k = 5
        ret,result,neigobours,dist = knn.findNearest(test,k = 1)

        #check whether there's a match
        matches = result == test
        correct = np.count_nonzero(matches)
        if correct == 1:
            outputText = Label(text='It\'s a number').pack()
        else:
            outputText = Label(text='It\'s a character').pack()

root = Uploader()
root.geometry("500x500+700+100")
root.title('"Number or Character" - Recognition Program')
root.mainloop()
```