from lib import \*

from implementation import \*

from visualization import \*

import io

import os

class main:

def \_\_init\_\_(self,master):

self.master = master

self.color\_fg = 'white'

self.color\_bg = 'gray1'

self.old\_x = None

self.old\_y = None

self.penwidth = 20

self.path="C:/Users/ashish agarwal/Desktop/pavan imp videos/MPR Project/drawn.png"

self.drawWidgets()

self.c.bind('<B1-Motion>',self.paint)#drawing the line

self.c.bind('<ButtonRelease-1>',self.reset)

self.black=(0,0,0)

self.width=500

self.height=500

self.image1 = PIL.Image.new("RGB", (self.width,self.height), self.black)

self.draw = ImageDraw.Draw(self.image1)

def paint(self,e):

if self.old\_x and self.old\_y:

self.c.create\_line(self.old\_x,self.old\_y,e.x,e.y,width=self.penwidth,fill=self.color\_fg,capstyle=ROUND,smooth=True)

self.draw.line([self.old\_x,self.old\_y,e.x,e.y],fill="white",width=self.penwidth)

self.old\_x = e.x

self.old\_y = e.y

def reset(self,e): #reseting or cleaning the canvas

self.old\_x = None

self.old\_y = None

def clear(self):

self.c.delete(ALL)

self.image1 = PIL.Image.new("RGB", (self.width,self.height),self.black)

self.draw = ImageDraw.Draw(self.image1)

def change\_fg(self): #changing the pen color

self.color\_fg=colorchooser.askcolor(color=self.color\_fg)[1]

def change\_bg(self): #changing the background color canvas

self.color\_bg=colorchooser.askcolor(color=self.color\_bg)[1]

self.c['bg'] = self.color\_bg

def change(self,event):

global path

if self.clicked.get()=='Upload Photo':

self.c.destroy()

self.c=None

self.path="C:/Users/ashish agarwal/Desktop/pavan imp videos/MPR Project/.png"

f=Frame(self.master,padx=10)

options=["1","2","3","4","5","6","7","8","9","0"

,"a","b","c","d","e","f","g","h","i","j","k","l","m","n","o","p"]

clicked=StringVar()

clicked.set(options[0])

drop=OptionMenu(f,clicked,options,command=img\_show)

drop.pack()

elif self.clicked.get()=='Draw Photo':

if not self.c:

self.c = Canvas(self.master,width=500,height=500,bg=self.color\_bg)

self.c.grid(row=0,column=3)

self.c.bind('<B1-Motion>',self.paint)

self.c.bind('<ButtonRelease-1>',self.reset)

self.path="C:/Users/ashish agarwal/Desktop/pavan imp videos/MPR Project/drawn.png"

def drawWidgets(self):

global output\_predict

options=[" ",

"Upload Photo",

"Draw Photo"

]

f1=Frame(self.master,padx=10,width=150)

self.clicked=StringVar(f1)

self.clicked.set(options[1])

drop=OptionMenu(f1,self.clicked,\*options,command=self.change)

drop.pack(pady=20)

f1.grid(row=0,column=0)

self.c = Canvas(self.master,width=500,height=500,bg=self.color\_bg)

self.c.grid(row=0,column=1)

menu = Menu(self.master)

self.master.config(menu=menu)

colormenu = Menu(menu)

menu.add\_cascade(label='Colors',menu=colormenu)

colormenu.add\_command(label='Brush Color',command=self.change\_fg)

colormenu.add\_command(label='Background Color',command=self.change\_bg)

optionmenu = Menu(menu)

menu.add\_cascade(label='Options',menu=optionmenu)

optionmenu.add\_command(label='Clear Canvas',command=self.clear)

optionmenu.add\_command(label='Exit',command=self.master.destroy)

self.f=Frame(self.master,padx=10,pady=10)

Button(self.f,text="clear",command=self.clear,width=50).grid(row=0,column=0)

Button(self.f,text="predict",width=50,command=self.output\_predict).grid(row=1,column=0)

Button(self.f,text="save",width=50,command=self.save\_img).grid(row=2,column=0)

self.f.grid(row=1,column=1)

def output\_predict(self):

op=img\_to\_array(self.path)

print("MY format",op)

print(len(op[0]))

y\_pred=predict\_output(X\_test=op,y\_test=[0])

print(chr(chartoascii[y\_pred[0]] ) )

Label(self.f,text="The predicted output is {}".format(chr(chartoascii[y\_pred[0]]))).grid(row=4,column=0)

def save\_img(self):

if self.c:

file="drawn.png"

self.image1.save(file)

root = Tk()

main(root)

root.title('Handwritten Character Recognition')

root.geometry("{0}x{1}+0+0".format(root.winfo\_screenwidth(), root.winfo\_screenheight()))

root.mainloop()