Final Project

February 8, 2020

In [ ]:
```
import zipfile
from PIL import Image
from PIL import ImageDraw
import pytesseract
import cv2 as cv
import numpy as np
import kraken
from kraken import pageseg
import string
import itertools
import threading
import time
import sys

data_structure = []
project_folder = 'readonly/'
small = 'small_img.zip'
large = 'images.zip'

def append_images():
    with zipfile.ZipFile(project_folder + large, 'r') as f:   #Open the Archive
        for name in f.namelist():  #For each file in the archive
            with f.open(name) as imageFile:  #Open the file
                img = Image.open(imageFile)  #Set the active image to the file
                testDictionary = {
                    "image" : img.copy(),
                    "boxes" : [],
                    "text" : [],
                    "face_boxes" : None,
                    "file_name" : name,
                    "order" : float(name[2:4])
                    }

                    data_structure.append(testDictionary)
```

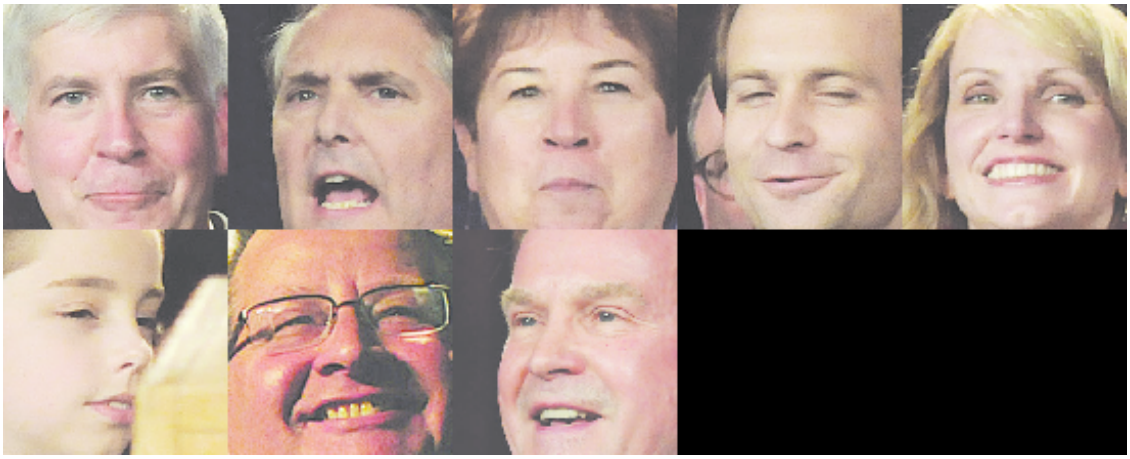
```
def text_detection():

    for item in data_structure:
        img = item['image']
        item['text'] = pytesseract.image_to_string(img).replace('-\n','')
        #boxes = item['boxes']



def face_detection():

    face_cascade = cv.CascadeClassifier(cv.data.haarcascades + 'haarcascade_frontalface
    eye_cascade = cv.CascadeClassifier(cv.data.haarcascades + 'haarcascade_eye.xml')

    for item in data_structure:
        cv_img = np.array(item['image'])
        cv_img_grey = cv.cvtColor(cv_img, cv.COLOR_BGR2GRAY)
        item['face_boxes'] = face_cascade.detectMultiScale(cv_img_grey, 1.30,4)  #.tol
    cv.destroyAllWindows()


def paste_to_contact_sheet():
    #Get test word from user
    test_word = None
    whichFile = None
    sorted_structure = sorted(data_structure, key = lambda i: i['order'])
    small_data_structure = sorted_structure[:4]
    while test_word != 'quit':
        test_word  = input("Enter a keyword: " + "\n" + "enter quit to quit" + "\n  ")
        if test_word == 'quit':
            break
        whichFile = input("Which file: 'small' or 'large' would you like to search)" +


        if whichFile == 'small':

#         if breaker != len(data_structure):
#             #for each image test to see if test_word is found in the text
            for item in small_data_structure:
                if test_word in item['text']:
                    print("Word found in {}".format(item['file_name']))
                    if item['face_boxes'] == ():
                        print("But there were no faces in that file")
                    #if [] in item['face_boxes'] == True:
                    #    print("But there were no faces in that file")
                    else:
                        img = item['image']
                        #Create a contact sheet
```

```
                        contact_sheet = Image.new(img.mode, (550,110*int(np.ceil(len(i
                        p = 0
                        q = 0
                        #For each box found in face boxes....
                    #extract the dimensions
                        box = item['face_boxes']
                        for x,y,w,h in box:
                            #set face equal to the face dimensions and crop from the o
                            face = img.crop((x,y,x+w,y+h)).resize((110,110))
                            face.thumbnail((110,110))
                            #display(face)
                            #paste the face to the contact sheet
                            contact_sheet.paste(face, (p, q))
                            #change x and y to fit the correct spot in the contact she
                            if p+110 == contact_sheet.width:
                                p=0
                                q=q+110
                            else:
                                p=p+110

                        display(contact_sheet)
                        #contact_sheet.show()


        elif whichFile == 'large':
#         if breaker != len(data_structure):
#             #for each image test to see if test_word is found in the text
            for item in sorted_structure:
                if test_word in item['text']:
                    print("Word found in {}".format(item['file_name']))
                    if item['face_boxes'] == ():
                        print("But there were no faces in that file")

                    #if [] in item['face_boxes'] == True:

                    else:
                        img = item['image']
                        #Create a contact sheet
                        contact_sheet = Image.new(img.mode, (550,110*int(np.ceil(len(i
                        p = 0
                        q = 0
                        #For each box found in face boxes....
                    #extract the dimensions
                        box = item['face_boxes']
                        for x,y,w,h in box:
                            #set face equal to the face dimensions and crop from the o
                            face = img.crop((x,y,x+w,y+h)).resize((110,110))
                            face.thumbnail((110,110))
```

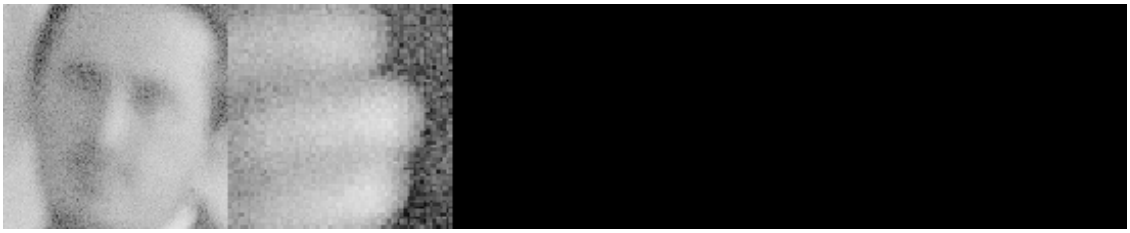
```
                                #display(face)
                                #paste the face to the contact sheet
                                contact_sheet.paste(face, (p, q))
                                #change x and y to fit the correct spot in the contact she
                                if p+110 == contact_sheet.width:
                                    p=0
                                    q=q+110
                                else:
                                    p=p+110

                            display(contact_sheet)
                            #contact_sheet.show()
                else:
                    print("Please enter the correct word: small or large")
                    continue

#                 else:
#                     breaker = breaker + 1
#                     print("word not found in {}".format(item['file_name']))
#                     print(breaker)
#
#         elif breaker >= len(data_strucutre):
#             print("that word is not found in any of the files")
#             print("try again")
#             continue


def animated_appending():
    chars = "/\|"
    for char in chars:
        sys.stdout.write('\r'+'    Appending...'+char)
        time.sleep(.1)
        sys.stdout.flush()

def animated_mining():
    chars = "/\|"
    for char in chars:
        sys.stdout.write('\r'+'    Mining...'+char)
        time.sleep(.1)
        sys.stdout.flush()

def main():
    # loading the face detection classifier
    #face_cascade = cv.CascadeClassifier('readonly/haarcascade_frontalface_default.xml
    # Creates the data structure to be used in this program
    append = threading.Thread(target=append_images)
    text_mining = threading.Thread(target=text_detection)
    face_mining = threading.Thread(target=face_detection)
```

```
            print("Building Data Structure")
            print("  Appending Images to Structure")
            #t = threading.Thread(target=animate)
            append.start()
            #append_images(data_structure)
            while append.isAlive():
                animated_appending()
            print("    Appended")


            print("  Mining images for text")
            text_mining.start()
            while text_mining.isAlive():
                animated_mining()
            print("    Mined")

            print("  Mining images for faces")
            face_mining.start()
            while face_mining.isAlive():
                animated_mining()
            print("    Mined")


            print("The Data Structure is complete")
            print("Get ready to search!")
            paste_to_contact_sheet()


        main()
```

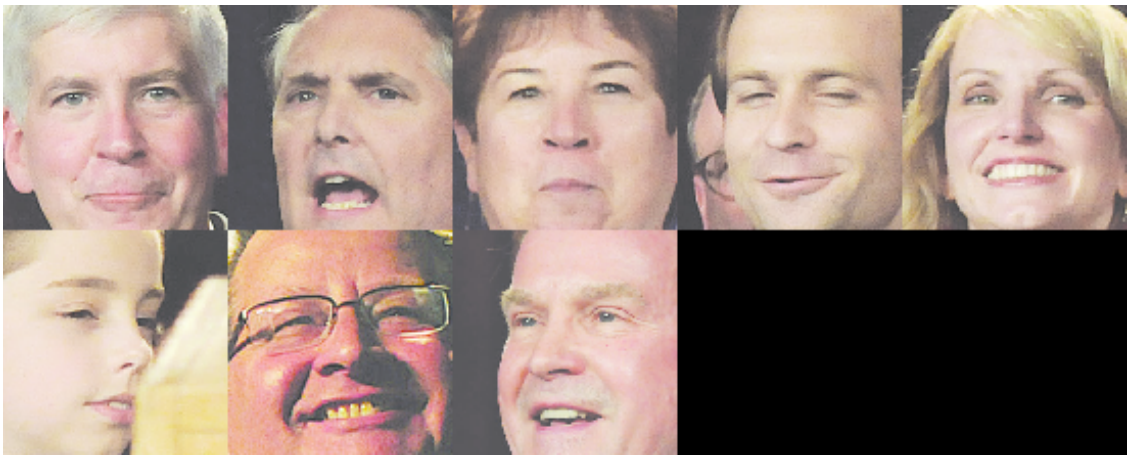
```
Building Data Structure
  Appending Images to Structure
    Appending...|    Appended
  Mining images for text
    Mining...|    Mined
  Mining images for faces
    Mining...|    Mined
The Data Structure is complete
Get ready to search!
Enter a keyword:
enter quit to quit
  Chris
Which file: 'small' or 'large' would you like to search)
  small
Word found in a-0.png
```



```
Word found in a-3.png
```



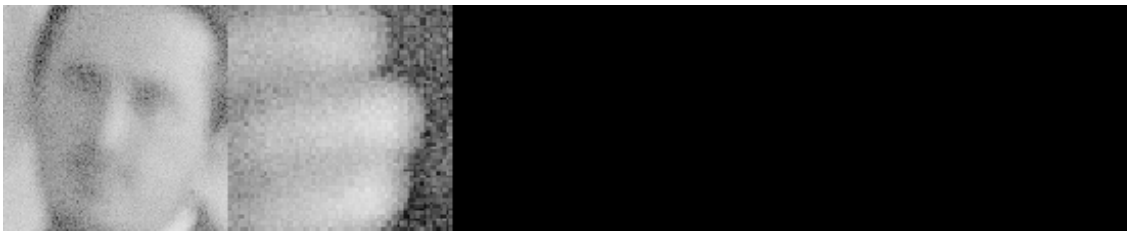
```
Enter a keyword:
enter quit to quit
  Mark
Which file: 'small' or 'large' would you like to search)
  large
Word found in a-0.png
```



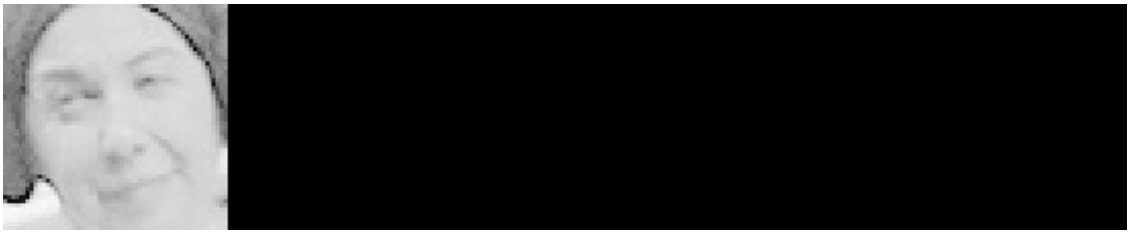
Word found in a-1.png

Word found in a-2.png

Word found in a-3.png



```
Word found in a-8.png
But there were no faces in that file
Word found in a-10.png
But there were no faces in that file
Word found in a-13.png
```



In [ ]: