COULON Siméon
SERMET-MAGDELAIN Alexandre

TP2 : Reconnaissance chiffres

I Préparation du TP

II Prise en main de l'environnement :

Nous avons effectué les actions données dans le TP.

III Travail préparatoire :

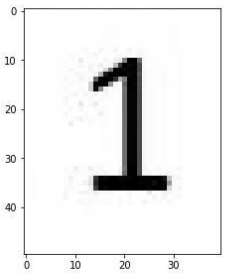
1) La classe image est définit par ces pixels, la largeur de l'image W et la longueur H. Ensuite il y a la méthode set_pixels qui permet d'affilier un tableau de 2 dimensions aux pixels. Les dernières méthodes permettent quant à elle d'afficher l'image.

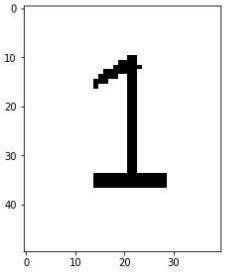
2) Pour écrire la fonction binarisation nous avons utilisé la méthode suivante :

```
def binarisation(self, S):
    im_bin = Image()
    im_bin.set_pixels(np.zeros((self.H, self.W), dtype=np.uint8))
    for i in range (0,self.H):
        for j in range (0,self.W):
            if self.pixels[i][j]>S:
                im_bin.pixels[i][j]=255

    return im_bin
```

Nous créons une nouvelle image de même taille que l'image que nous allons binariser. Puis on parcourt le tableau de pixel de cette image en comparant chacune des valeurs à notre paramètre S. la valeur du pixel est plus grande que notre paramètre alors on change la valeur de ce pixel avec la valeur 255 (qui correspond à blanc). Sinon on ne change pas la valeur car notre nouveau tableau de pixel est initialisé pour des valeurs de pixel égal à 0.

```
In [25]: runfile('E:/alex/tpinfo2/tp2-reconnaissance-chiffres-tp2_coulon_sermet-magdelain-main/src/main.py', wdir='E:/alex/
tpinfo2/tp2-reconnaissance-chiffres-tp2_coulon_sermet-magdelain-main/src')
Reloaded modules: image, reconnaissance
lecture image : ../assets/test2.JPG (50x40)
```





Dans cet exemple S=70

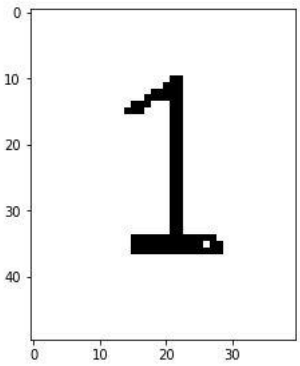
3) Pour la méthode localisation le code est :

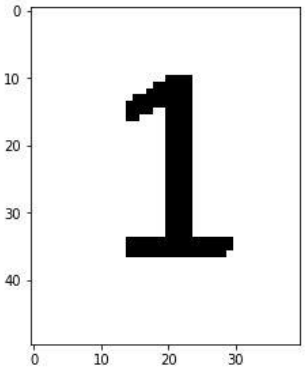
```
def localisation(self):
    im_loc = Image()

    l_min = self.H
    c_min = self.W
    l_max = 0
    c_max = 0
    for i in range (0,self.H):
        for j in range (0,self.W):
            if self.pixels[i][j]==0:
                if j<c_min:
                    c_min=j
                elif j>c_max:
                    c_max=j
                elif i<l_min:
                    l_min=i
                elif i>l_max:
                    l_max=i
    im_loc.set_pixels(self.pixels[l_min:l_max+1, c_min:c_max+1])

    return im_loc
```

Tout d'abord on crée une nouvelle image et les 4 paramètres permettant de recréer l'image localisée. On parcourt le tableau de pixel. Dès qu'un pixel est noir on regarde ces cordonnées. Le but de notre code est d'avoir une valeur de c_min et l_min la plus grande possible. De même que l'on veut la valeur la plus grande possible pour c_max et l_max. Pour chaque pixel noirs si ces cordonnées sont plus grand que c_max/lmax alors il prend sa valeur ou si ces cordonnes sont inférieur à c_min/l_min alors il prend sa valeur.

IV Reconnaissance automatique de chiffre

1) Résultats pour S=10 et pour S=200,

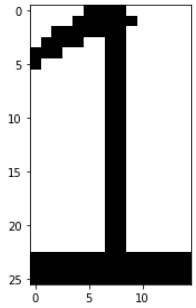


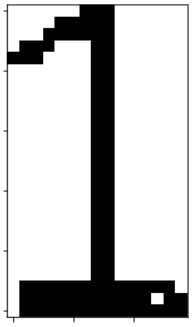


On remarque que le deuxième est plus gras. En effet on augmente le seuil, donc il aura logiquement plus pixels qui vont passer en noir totale.

COULON Siméon
SERMET-MAGDELAIN Alexandre

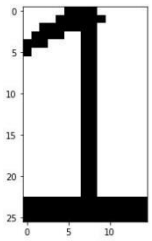
2) Résultats pour S=70 et pour S=10, la localisation est différente mais fonctionne dans les 2 cas.

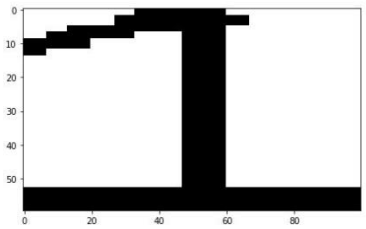




3)Pour cette fonction nous avons utlisé une fonction importé de la librairie skymage. Notre code est la suivant :

```
def resize(self, new_H, new_W):
    im_res= Image()
    im_1=resize(self.pixels, (new_H,new_W), 0)
    im_res.set_pixels(np.uint8(im_1*255))
    return im_res
```





Notre résultats est le suivant pour une redimensionné à 60 de haut et 100 de large.

Nous avons tout d'abord créé une image que nous allons retourner. Puis nous modifions le tableau de pixels de 'self' grâce à la fonction resize. Attention la fonction resize renvoie un tableau. Puis dans un second temps nous remodifions ce tableaux avec la commande 'np.uint8(pixels_resized*255)'. Enfin nous affectons ce tableau final au tableau pixel de notre image créé au départ grâce à la méthode set_pixel.

4) Voici le code pour la méthode similitude :

```
def similitude(self, im):
    npixels= self.H*self.W
    cpt = 0


    for i in range (0,self.H):
        for j in range (0,self.W):
            if self.pixels[i][j]==im.pixels[i][j]:
                cpt +=1

    return cpt/npixels
```

Pour comparer les deux images de même taille, on calcule le nombre de pixels constituant les images et on crée un compteur. On parcourt tout le tableau et augmente le compteur si les valeurs des pixels des deux images sont égales.

On retourne une valeur de similitude en faisant le quotient du compteur avec le nombre de pixels.

Après avoir fini toutes les méthodes dans le fichier image, on vérifie que tous les tests sont corrects :

```
In [120]: runfile('E:/alex/tpinfo2/tp2-reconnaissance-chiffres-
tp2_coulon_sermet-magdelain-main/tests/test_Image.py', wdir='E:/alex/tpinfo2/
tp2-reconnaissance-chiffres-tp2_coulon_sermet-magdelain-main/tests')
.........Reloaded modules: image, reconnaissance
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test5.JPG (54x61)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test2.JPG (50x40)
lecture image : ../assets/test4.JPG (58x40)
lecture image : ../assets/test3.JPG (43x38)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test1.JPG (54x43)
.............lecture image : ../assets/test2.JPG (50x40)
lecture image : ../assets/test4.JPG (58x40)
lecture image : ../assets/test3.JPG (43x38)
lecture image : ../assets/test2.JPG (50x40)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test2.JPG (50x40)
lecture image : ../assets/test2.JPG (50x40)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test3.JPG (43x38)
lecture image : ../assets/test2.JPG (50x40)

----------------------------------------------------------------------
Ran 22 tests in 0.282s

OK
An exception has occurred, use %tb to see the full traceback.

SystemExit: False

C:\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3304:
UserWarning: To exit: use 'exit', 'quit', or Ctrl-D.
  warn("To exit: use 'exit', 'quit', or Ctrl-D.", stacklevel=1)
```

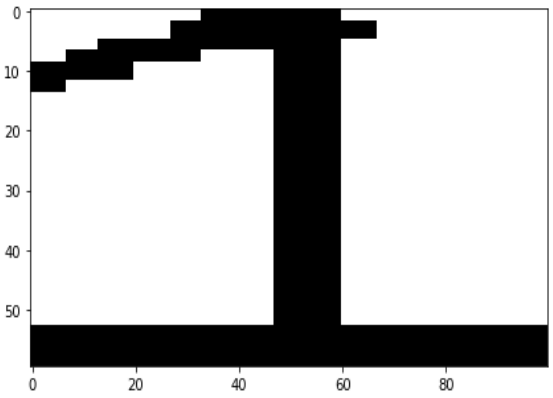
COULON Siméon
SERMET-MAGDELAIN Alexandre

5) Voici le code de notre méthode reconnaissance :

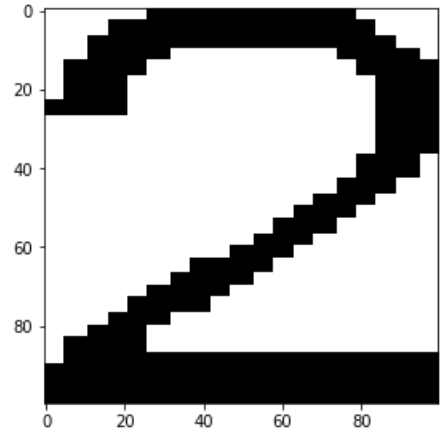
```
def reconnaissance_chiffre(image, liste_modeles, S):
    res=0
    cpt=-1
    A=image.binarisation(S)
    b=A.localisation()
    for i in list (liste_modeles):
        c=b.resize(i.H, i.W)
        if c.similitude(i)>res:
            res=c.similitude(i)
            cpt=cpt+1
    return cpt
```

Tout d'abord on initialise un résultat égal à 0. On initialise un compteur a -1 car nous parcourons au minimum une boucle, ce qui lui donnera la plus faible valeur possible soit 0. A travers les différentes méthodes créées en amont on transforme notre image de façon à avoir une similitude la plus correct possible. Ensuite on parcourt notre liste d'images qui sont toutes comparées à notre image. A chaque fois que la similitude entre notre image et l'image i de la liste le resautas de la similitude prend la place de res et le compteur incrémenté de 1. En fin de boucle on revoie la valeur du compteur qui correspond au numéro ayant le plus de similitude.

Les résultats sont :



```
lecture image : ../assets/_0.png (32x22)
lecture image : ../assets/_1.png (32x18)
lecture image : ../assets/_2.png (32x20)
lecture image : ../assets/_3.png (32x20)
lecture image : ../assets/_4.png (32x24)
lecture image : ../assets/_5.png (32x20)
lecture image : ../assets/_6.png (32x21)
lecture image : ../assets/_7.png (32x21)
lecture image : ../assets/_8.png (32x22)
lecture image : ../assets/_9.png (32x22)
Le chiffre reconnu est :  1
```



```
lecture image : ../assets/_0.png (32x22)
lecture image : ../assets/_1.png (32x18)
lecture image : ../assets/_2.png (32x20)
lecture image : ../assets/_3.png (32x20)
lecture image : ../assets/_4.png (32x24)
lecture image : ../assets/_5.png (32x20)
lecture image : ../assets/_6.png (32x21)
lecture image : ../assets/_7.png (32x21)
lecture image : ../assets/_8.png (32x22)
lecture image : ../assets/_9.png (32x22)
Le chiffre reconnu est :  2
```

On remarque que pour certain chiffre la reconnaissance automatique ne fonctionne pas très bien.

Puis on remarque dans un second temps que changer la valeur de bancarisation n'impact pas beaucoup voir pas du tout la reconnaissance.

COULON Siméon
SERMET-MAGDELAIN Alexandre

Conclusion :

Lors de ce TP nous avons pu nous familiariser avec les classes et les matrices. La fonction localisée nous a posé un peu plus de problème que les autres mais nous avons réussi à trouver nos erreurs.