Rakotondrainibe Thomas
Ligonnière Nino
02/12/2021

Rapport de TP2 – Lecture automatique de chiffres par analyse d'image

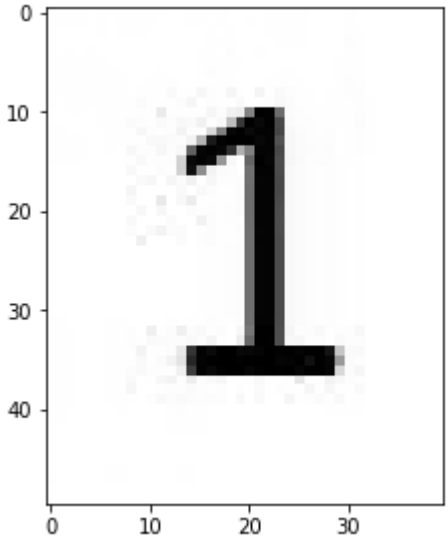
I. Introduction

Le TP porte sur la représentation et la manipulation des images numériques en noir et blanc sur Python.

II. Section 1 du TP

1. Question (1).

L'image s'affiche correctement dans la console



2. Question (2).

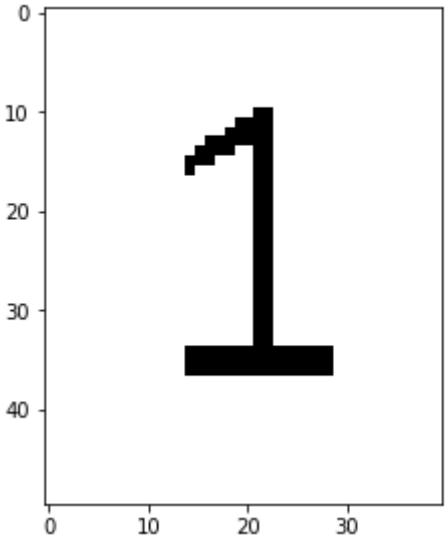
On binarise l'image avec le code suivant :

```
def binarisation(self, S):
    # creation d'une image vide
    im_bin = Image()

    # affectation a l'image im_bin d'un tableau de pixels de meme taille
    # que self dont les intensites, de type uint8 (8bits non signes),
    # sont mises a 0
    im_bin.set_pixels(np.zeros((self.H, self.W), dtype=np.uint8))
    for i in range(self.H):
        for k in range(self.W):
            if self.pixels[i,k] < S:
                im_bin.pixels[i,k] = 0
            else:
                im_bin.pixels[i,k] = 255
    # TODO: boucle imbriquees pour parcourir tous les pixels de l'image im_bin
    # et calculer l'image binaire

    return im_bin
```

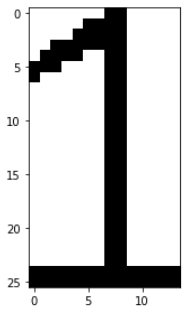
Cela renvoit bien l'image en noir et blanc sans niveau de gris lorsqu'on teste dans le main



3. Question (3).

```
def localisation(self):
    im_loc = Image()
    c_min = self.W
    c_max = 0
    l_min = self.H
    l_max = 0
    for i in range(self.H):
        for j in range(self.W):
            if self.pixels[i][j] == 0:
                if i < l_min:
                    l_min = i
                if i > l_max:
                    l_max = i
                if j < c_min:
                    c_min = j
                if j > c_max:
                    c_max = j
    im_loc.set_pixels(self.pixels[l_min:l_max,c_min:c_max])
    return im_loc
```

cela nous sort l'image relocalisé en lançant le main



III. Reconnaissance automatique de chiffre

1. Question (1)

Plus le seuil est élevé plus on retrouve de pixel noir autour du chiffre
En revanche plus il est bas plus il y a de pixels blancs dans le chiffre

2. Question(2)

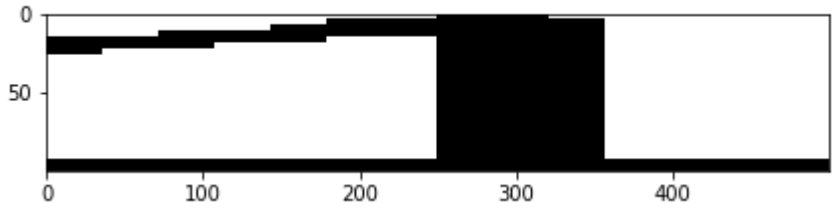
On a essayé avec plusieurs seuil et l'image relocalisée est plus grande lorsque des pixels noirs apparaissent à l'extérieur du chiffre.

3. Question (3)

On code la fonction resize :

```
#=============================================================================
# Methode de redimensionnement d'image
#=============================================================================
def resize(self, new_H, new_W):
    im_resized = Image()
    im_resized.set_pixels(np.uint8(resize(self.pixels , (new_H,new_W) , 0)*255))
    return im_resized
```

L'image relocalisé une fois redimensionné nous donne :



Pour des valeurs de new_H et new_W de 100 et 500

4. Question(4)

On code la fonction similitude() :

```
#=============================================================================
# Methode de mesure de similitude entre l'image self et un modele im
#=============================================================================
def similitude(self, im):
    pixels_commun = 0
    if self.H == im.H and self.W == im.W:
        for i in range(self.H):
            for k in range(self.W):
                if self.pixels[i,k] == im.pixels[i,k]:
                    pixels_commun += 1
    return pixels_commun/(self.H*self.W)
```

On test avec un print(image.similitude(image_binarisee)) pour comparer les deux est la console nous sort 0.565.

5. Question (5)

On code la fonction reconnaissance_chiffre :

```
def reconnaissance_chiffre(self , liste_modeles , S):
    im_bin = self.binarisation(S)
    im_local = im_bin.localisation()
    sim = 0
    res=0
    for i in range(len(liste_modeles)):
        modele = liste_modeles[i]
        im_resized = im_local.resize(modele.H , modele.W)
        sim_i = im_resized.similitude(modele)
        if sim_i > sim:
            sim = sim_i
            res = i
    return res
```

Le chiffre reconnu par la fonction reconnaissance_chiffre est bien le 1

6. Question(6)

On a testé avec différents chiffre et à chaque fois toutes les fonctions marchaient.

IV. Conclusion

Nous sommes arrivés tant bien que mal à la fin du TP.
On a rencontré de grosse difficulté sur la méthode relocalisation car nous n'arrivions pas à trouver comment avoir les bornes inferieurs.
Nous avons découvert ce qu'était une image en python et comment les gérer.