LECORVAISIER Elias
VERMOT DESROCHES Léo
03/12/2021

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

I. Introduction

L'objectif du TP2 est de programmer un code permettant la reconnaissance d'un chiffre à partir d'une image. Pour ce faire, nous allons créer de nombreuses fonctions permettant de construire notre programme final étape par étape

II. Travail préparatoire

1. Question (2).

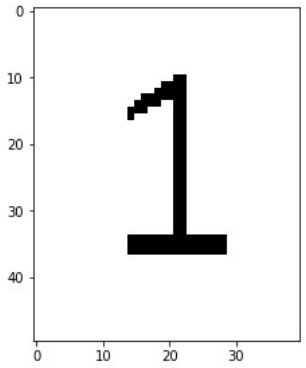
Dans cette question, on nous demande de créer un code python permettant de simplifier une image et de faire ressortir l'information utile. Pour ce faire, nous rendons les pixels d'intensité inférieure à S noirs et le reste en blanc.

Nous créons donc la méthode **binarisation** :

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

La valeur S est une valeur modifiable.

```
S = 70
image_binarisee = image.binarisation(S)
image_binarisee.display("Image binarisee")
```



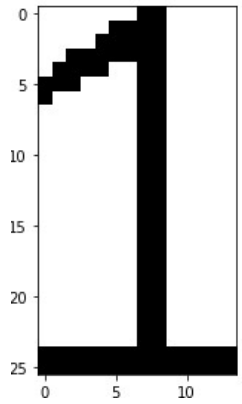
Résultat avec S=70

2. Question (3).

Nous devons ensuite rogner l'image sur le chiffre afin de supprimer les vides inutiles. Pour ce faire, nous avons créé la méthode **localisation**, dans laquelle nous cherchons les bordures lmin, lmax, cmin, cmax, puis nous les insérons en tant que nouvelles bordures.

```
def localisation(self):
    l_min = self.H
    l_max = 0
    c_min = self.W
    c_max = 0
    im_loc = Image()
    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
```

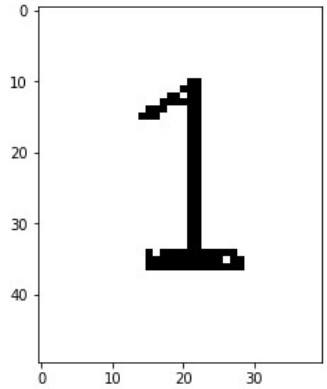
```
image_localisee = image_binarisee.localisation()
image_localisee.display("Image localisee")
```



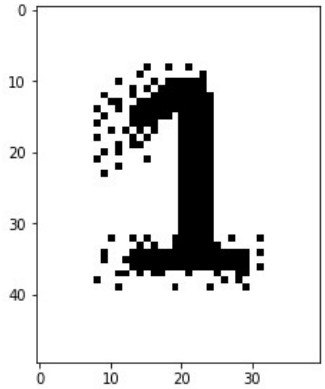
Résultat de la méthode

III. Reconnaissance automatique de chiffre

1. Question (1).



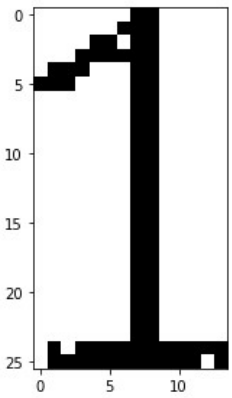
Résultat avec S= 10



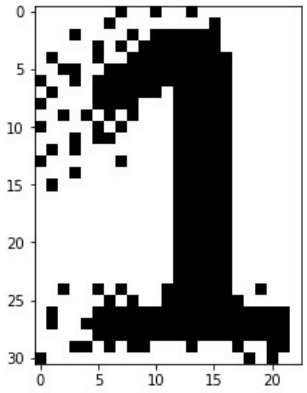
Résultat avec S= 250

Le module nose2 n'est pas installé sur l'ordinateur, le test sur GitHub n'indique aucune erreur.

2. Question (2).



Résultat avec S= 10



Résultat avec S= 250

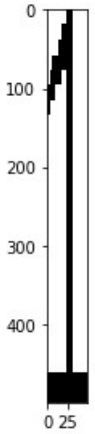
Le module nose2 n'est pas installé sur l'ordinateur, le test sur GitHub n'indique aucune erreur.

3. Question (3).

On cherche dans cette question à redimensionner notre chiffre afin de l'adapter à la taille de différents modèles envisagés.

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


image_resizee = image_localisee.resize(500, 50)
image_resizee.display("Image redimensionee")
```



Resultat pour S=70, newH=500 et newW=50

Le module nose2 n'est pas installé sur l'ordinateur, le test sur GitHub n'indique aucune erreur.

4. Question (4).

Pour cette question, l'objectif est de déterminer la similitude entre deux images. Plus le résultat est proche de 1, plus les images sont semblables et inversement.

```
def similitude(self, im):
    pix_id = 0
    pix_diff = 0
    for i in range (self.H):
        for j in range (self.W):
            if self.pixels[i,j] == im.pixels[i,j]:
                pix_id +=1
            else:
                pix_diff +=1
    return pix_id/(pix_id + pix_diff)
```

5. Question (5).

Enfin, nous avons créé la méthode permettant de retrouver quel est le chiffre dans l'image.

```
def reconnaissance_chiffre(image, liste_modeles, S):
    sim_max = 0
    id_sim_max = 0
    im_bin = image.binarisation( S)
    im_loc = im_bin.localisation()
    for i in range (len(liste_modeles)):
        im_redim = im_loc.resize(liste_modeles[i].H, liste_modeles[i].W)
        sim = im_redim.similitude(liste_modeles[i])
        if sim > sim_max :
            sim_max = sim
            id_sim_max = i
    return id_sim_max

liste_modeles = lecture_modeles(path_to_assets)
chiffre = reconnaissance_chiffre(image, liste_modeles, 70)
print("Le chiffre reconnu est : ", chiffre)
```

```
Le chiffre reconnu est :  1
```

Le code fonctionne donc correctement.

6. Question (6).

N° du Test	Seuil	Véritable nombre	Nombre trouvé par le code
Test 1	70	4	4
	200		4
Test 7	70	5	7
	10		7
Test 8	70	1352	7
	200		7
Test 9	70	18456	7
	10		7
Test 10	70	6	6
	200		6

IV. Conclusion

Nous avons finalement réussi à programmer un code permettant de trouver un chiffre à partir d'une image. Au cours de ce TP, nous avons rencontré quelques difficultés, notamment, concernant le regroupement de plusieurs codes créés au préalable. Cependant, nous avons également appris à utiliser les fonctions test.

Pour conclure, le programme que nous avons créé fonctionne correctement pour les chiffre hormis ceux ayant des polices particulières. En revanche, le programme n'est pas adapté pour les nombres.