DELICOURT Raphaël
DELESALLE Thibault
02/12/2021

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

I. Introduction

Ici, nous allons étudier des images numériques, représentées par des tableaux à deux dimensions. Chaque élément du tableau est un nombre compris entre 0 et 255 et symbolise l'intensité du pixel.

Dans ce TP, nous allons nous intéresser à la reconnaissance par corrélation avec des modèles.

II. Travail préparatoire

1. Question 1

La classe image permet la création des attributs H, W et pixels qui représentent respectivement la hauteur (H=high) et la largeur (W=wide) de l'image. Ce sont les nombres de pixels en hauteur et en largeur. L'attribut pixels est un tableau qui contient une valeur entre 0 et 255. 0 est associé au noir et 255 est associé au blanc.

2. Question 2

Dans cette fonction, on utilise deux boucles for pour parcourir tous les pixels de l'image. Si la valeur du pixel est inférieure au seuil choisi, le pixel devient noir (valeur 0). Sinon, le pixel devient blanc (valeur 255).

```
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]=0
            else:
                im_bin.pixels[i][j]=255
    return im_bin
```

3. Question 3

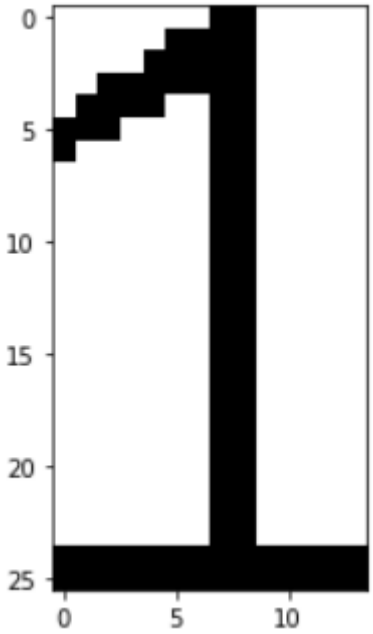
Tout d'abord, on initialise c_min à la valeur maximum en largeur et l_min à la valeur maximum de hauteur. Puis on initialise c_max et l_max à 0, la valeur minimum.

Ensuite, on utilise deux boucles for pour parcourir toutes les cases du tableau. On compare les valeurs pour les quatre paramètres et on les remplace quand on rencontre un pixel noir.

Enfin, on retourne l'image qui est bien constituée d'un rectangle encadrant la forme noire (sans bords blancs autour)

```python
def localisation(self):
    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:
                l_max=i
            if (self.pixels[i,j]==0 and i<l_min):
                l_min=i
            if (self.pixels[i,j]==0 and j<c_min):
                c_min=j
            if (self.pixels[i,j]==0 and j>c_max):
                c_max=j
    new_Image=Image()
    new_Image.set.pixels(self.pixels[l_min:l_max,c_min:c_max])
    return new_Image
```

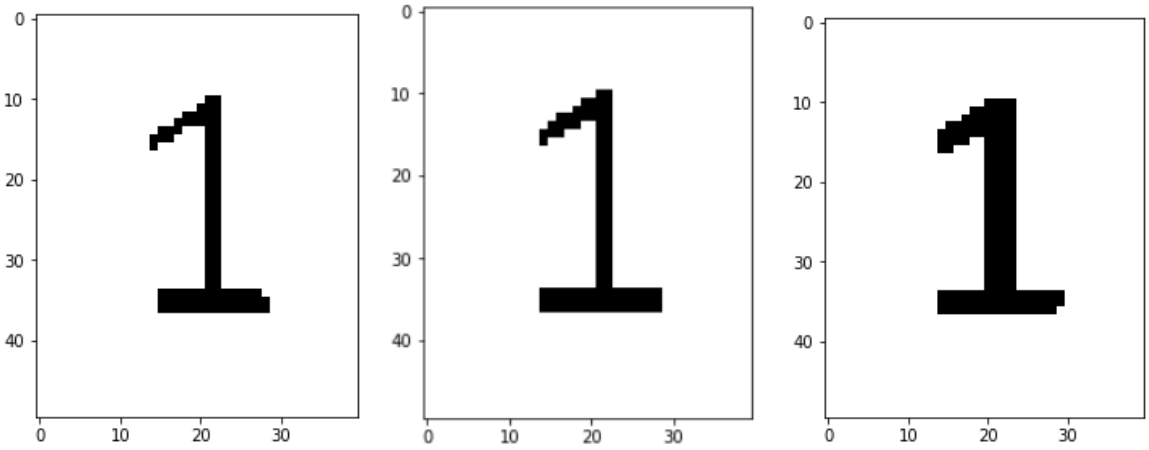
Le test fonctionne.



III. Reconnaissance automatique de chiffre

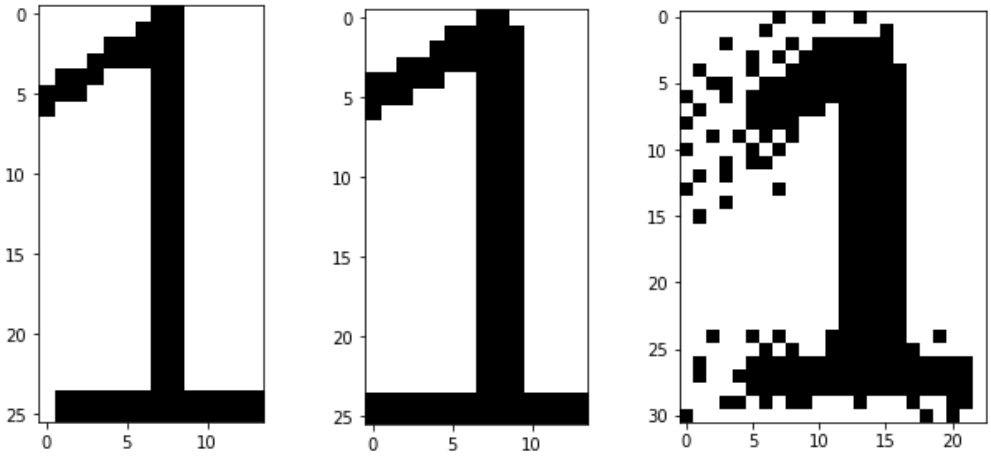
1. Question 1

La méthode binarisation fonctionne pour différentes valeurs de seuil. Ci -dessous, la première image correspond à S=20, la deuxième à S=70 et la troisième à S=200.



2. Question 2

Cela fonctionne pour différentes valeurs de seuil : les images renvoyées sont bien des cadres englobant les formes noires. La première image correspond à un seuil S=20, la deuxième à S=100 et la troisième à S=250.

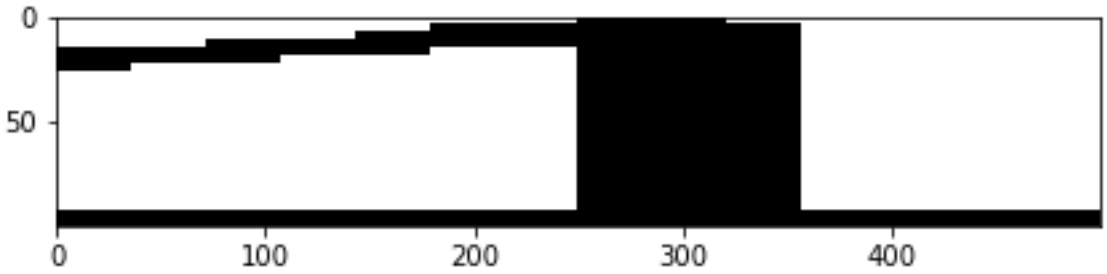


3. Question 3

Pour la fonction resize, on crée une variable tab qui stocke les valeurs des pixels. Le résultat de la fonction resize est codé sur une échelle en float et non plus en int comme précédemment dans le TP. Pour cela, on utilise np.uint8(pixels_resized*255). Enfin, on convertit l'image et on la renvoie.

```
def resize(self, new_H, new_W):
    image3=Image()
    tab=resize(self.pixels,(new_H,new_W),0)
    tab=np.uint8(tab*255)
    image3.set_pixels(tab)
    return image3
```

Après le test, on constate que les dimensions de l'image on bien changé. Nous avons les dimensions (100,500).



4. Question 4

Dans la fonction similitude, on crée un compteur appelé sim. Puis, on parcourt le tableau pixel par pixel et on compare leur intensité au même pixel pour les 2 images. Si elle est identique, on ajoute 1 au compteur. Enfin, on retourne le compteur divisé par le nombre total de pixels ce qui nous donne un résultat compris entre 0 et 1, 1 étant la ressemblance parfaite (images identiques à 100%).

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

5. Question 5

Pour la fonction reconnaissance_chiffre, on commence par binariser l'image passée en paramètre avec la fonction binarisation. Puis on localise cette image avec la fonction localisation. On procède ensuite à une boucle qui parcourt toutes les images de la liste. A l'intérieur de la boucle, on redimensionne l'image à la taille du modèle avec la fonction resize et on calcule sa similitude. Enfin, on compare les indices de similitudes. Si l'indice calculé est supérieur à l'indice précédent, on le remplace. On retourne sim_tot, un nombre compris entre 0 et 9, qui correspond au rang de l'image avec l'indice de similitude le plus élevé.

```python
from image import Image

def lecture_modeles(chemin_dossier):
    fichiers= ['_0.png','_1.png','_2.png','_3.png','_4.png','_5.png','_6.png',
            '_7.png','_8.png','_9.png']
    liste_modeles = []
    for fichier in fichiers:
        model = Image()
        model.load(chemin_dossier + fichier)
        liste_modeles.append(model)
    return liste_modeles


def reconnaissance_chiffre(image, liste_modeles, S):
    image_bin=image.binarisation(S)
    image_loc=image_bin.localisation()
    sim_max=0
    sim_tot=0
    for i in range (0,len(liste_modeles)):
        image4=image_loc.resize(liste_modeles[i].H,liste_modeles[i].W)
        sim=image4.similitude(liste_modeles[i])
        if sim>sim_max:
            sim_max=sim
            sim_tot=i
    return sim_tot
```

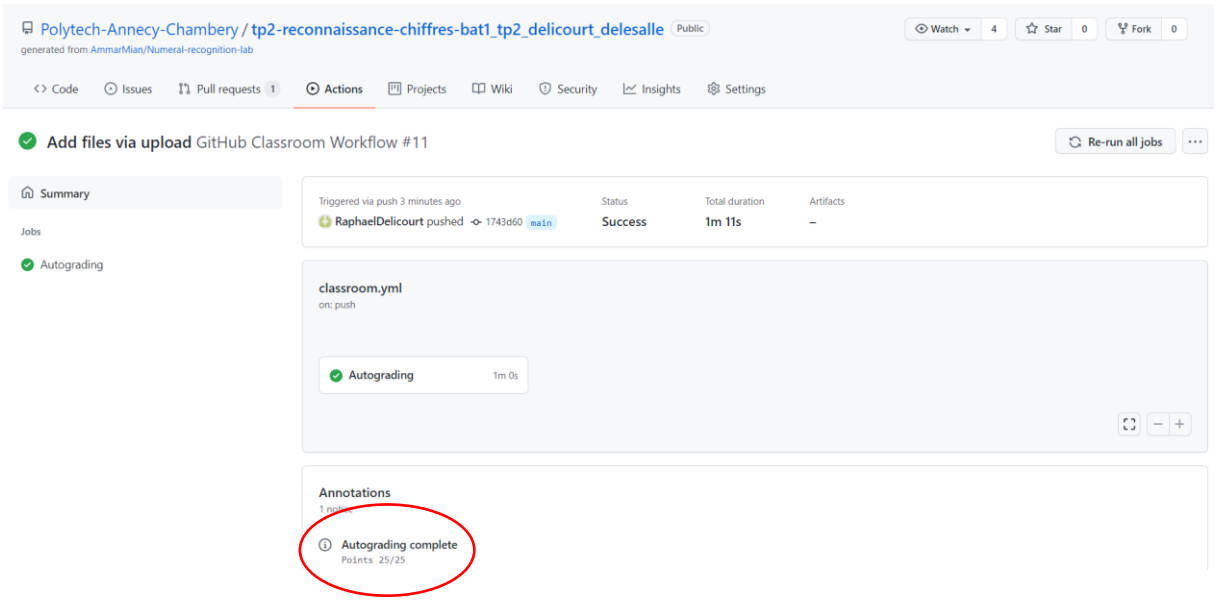
6. Question 6

Nous avons procédé à plusieurs tests en changeant les fichiers tests et les seuils dans le fichier main.py. Voici les résultats :

image	seuil	resultat
test4	100	2
test5	150	2
Test6	200	4
Test7	200	5
Test8	100	1 3 5 2

IV. Conclusion

Ce TP nous a permis de comprendre comment fonctionne la reconnaissance d'image par corrélation. Le main ne présente pas d'erreurs d'après les test Github.


Polytech-Annecy-Chambery / tp2-reconnaissance-chiffres-bat1_tp2_delicourt_delesalle
Public
generated from AmmarMian/Numeral-recognition-lab
Watch 4
Star 0
Fork 0
Code
Issues
Pull requests 1
Actions
Projects
Wiki
Security
Insights
Settings
Add files via upload GitHub Classroom Workflow #11
Re-run all jobs
Summary
Jobs
Autograding
Triggered via push 3 minutes ago
RaphaelDelicourt pushed 1743d60 main
Status
Success
Total duration
1m 11s
Artifacts
–
classroom.yml
on: push
Autograding 1m 0s
Annotations
Autograding complete
Points 25/25