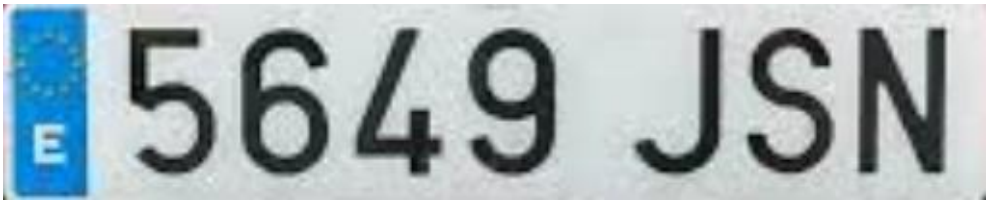
VC: Informe de Laboratori 8

Classificació d'objectes binaris amb descriptors



Pere Ginebra Solanellas

25/4/2021 – Q2 Curs 2020-21

Visió per Computador, FIB UPC

1. Introducció

En aquesta sessió de laboratori classificarem objectes d'una imatge binària segons les seves característiques o propietats per tal de poder-los identificar en altres imatges. Per això utilitzarem tant *bwconncomp* per identificar els elements individuals, *regionprops* per analitzar les seves característiques i *predict* per classificar-los. En concret intentarem reconèixer els caràcters d'una matricula de cotxe.

2. Exercici

Per començar, llegim la imatge base, la passem a escala de grisos i la binaritzem. Per obtenir les propietats de cada caràcter extreiem els components connexos amb *bwconncomp*:

```
I = rgb2gray(imread('Joc_de_caracters.jpg'));
BI = I < 128;
imshow(BI)
CC = bwconncomp(BI);
```

Figura 2.1. lectura de la imatge inicial i anàlisis dels components connexos

Fet això, evaluem les propietats de cada component/caràcter amb regionprops. En concret he decidit extreure: les longituds de els eixos menor i major, la soliditat (relació bounding-box/àrea), l'extent (relació convex-hull/àrea), la circularitat, l'eccentricitat, el centre de masses i la bounding-box de cada caràcter:

```
props = regionprops(CC,'MajoraxisLength','MinoraxisLength','Solidity','Extent','Circularity','Eccentricity','Centroid','BoundingBox');
```

Figura 2.2. obtenció de propietats de cada component/caràcter

A continuació utilitzo aquests valors per a generar un vector de propietats per cada caràcter. Amb els eixos calculo un ratio entre el menor i el major (perquè es mantingui encara que el caràcter es re-escali) i amb el centre de masses i la bounding box calculo el centre de masses relatiu de cadascun, ja que el *Centroid* dona el centre de masses posicionat en la imatge. La resta de propietats les afegeixo al vector X intactes:

```
ratio = [props(:).MinorAxisLength]./[props(:).MajorAxisLength];
centroids = cat(1, props.Centroid);
boxes = cat(1,props.BoundingBox);
centroid_x1 = ([centroids(:,1)]'-[boxes(:,1)]');
centroid_y1 = ([centroids(:,2)]'-[boxes(:,2)]');
rel_cent_x1 = centroid_x1./[boxes(:,3)]';
rel_cent_y1 = centroid_y1./[boxes(:,4)]';

X = [ratio; props.Solidity; props.Extent; props.Eccentricity; props.Circularity; rel_cent_x1; rel_cent_y1]';
```

Figura 2.3. Generació del vector de característiques

Aquestes característiques identifiquen cada caràcter per la seva densitat de píxels en relació a la seva mida (*solidity* i *extent*), la distribució d'aquests ("rel_cent_x/y") i la seva forma (*circularity,* "ratio" i *eccentricity*).

Finalment inicialitzo un vector per donar nom a cada caràcter i genero el classificador amb *TreeBagger:*

```
OUT = {'0' '1' '2' '3' '4' '5' '6' '7' '8' '9' 'B' 'C' 'D' 'F' 'G' 'H' 'J' 'K' 'L' 'M' 'N' 'P' 'R' 'S' 'T' 'V' 'W' 'X' 'Y' 'Z'};

Classifier = TreeBagger(100, X, OUT);
```

Figura 2.4. Generació del classificador

Ara només cal repetir les operacions anteriors sobre la imatge a avaluar i cridar la funció *predict* sobre els vectors de propietats dels elements de la nova imatge amb el classificador prèviament calculat. Els resultats obtinguts es poden representar utilitzant la funció *table:*

```
[label,score] = predict(Classifier, X);
table(Classifier.ClassNames, label, max(score)', 'VariableNames',{'Name','Label','Score'})
```

Figura 2.5. Anàlisis de la imatge problema i representació dels resultats

Per provar el script sobre matricules de veritat simplement he substituït les anteriors línies per aquestes:

```
matricula = ['5';'6';'4';'9';'J';'S';'N'];
%matricula = ['2';'0';'9';'3';'G';'S';'W'];
X_mat = [X(:,:); X(:,:); X(:,:); X(:,:); X(1:2,:)];

[label,score] = predict(Classifier, X_mat);
label1 = label(1:7);
score1 = score(1:7,:)';
m = max(score1)';
table(matricula, label1, m, 'VariableNames',{'Name','Label','Score'})
```

Figura 2.6. Anàlisis d'una matricula real amb 7 caràcters

Que asseguren que els vectors que es comparen tenen la mateixa mida (ja que les matricules tenen 7 caràcters i la nostra imatge base 30) i que l'output utilitza els labels correctes per veure la predicció de la matricula seleccionada.

3. Resultats

El resultat obtingut al analitzar la imatge “Joc_de_caracters_deformats.jpg” és el següent:

Name	Label	Score	{'9'}	{'9'}	0.55	{'P'}	{'P'}	0.63
			{'B'}	{'B'}	0.38	{'R'}	{'R'}	0.48
			{'C'}	{'C'}	0.53	{'S'}	{'S'}	0.45
{'0'}	{'0'}	0.61	{'D'}	{'D'}	0.58	{'T'}	{'T'}	0.48
{'1'}	{'1'}	0.51	{'F'}	{'F'}	0.44	{'V'}	{'V'}	0.27
{'2'}	{'2'}	0.19	{'G'}	{'G'}	0.58	{'W'}	{'W'}	0.51
{'3'}	{'3'}	0.4	{'H'}	{'H'}	0.51	{'X'}	{'X'}	0.26
{'4'}	{'4'}	0.49	{'J'}	{'J'}	0.63	{'Y'}	{'Y'}	0.47
{'5'}	{'5'}	0.39	{'K'}	{'K'}	0.57	{'Z'}	{'Z'}	0.52
{'6'}	{'6'}	0.52	{'L'}	{'L'}	0.61			
{'7'}	{'7'}	0.57	{'M'}	{'M'}	0.36			
{'8'}	{'8'}	0.61	{'N'}	{'N'}	0.41			

Figura 3.1. Resultat d’analitzar la imatge *Joc_de_caracters_deformats.jpg*

Com podem veure la majoria de prediccions tenen un puntuació propera al 0.4 – 0.5, però hi ha alguns caràcters que li costen més, com per exemple el ‘2’, amb una puntuació més propera a 0.2.

Els resultat obtingut al analitzar la imatge “Joc_de_caracters_deformats II.png” és el següent:

Name	Label	Score	{'9'}	{'W'}	0.17	{'P'}	{'8'}	0.33
			{'B'}	{'B'}	0.21	{'R'}	{'W'}	0.16
			{'C'}	{'B'}	0.06	{'S'}	{'W'}	0.08
{'0'}	{'0'}	0.21	{'D'}	{'0'}	0.09	{'T'}	{'P'}	0.11
{'1'}	{'1'}	0.36	{'F'}	{'P'}	0.14	{'V'}	{'W'}	0.16
{'2'}	{'V'}	0.13	{'G'}	{'6'}	0.02	{'W'}	{'W'}	0.24
{'3'}	{'9'}	0.08	{'H'}	{'M'}	0.13	{'X'}	{'6'}	0.04
{'4'}	{'4'}	0.16	{'J'}	{'4'}	0.12	{'Y'}	{'P'}	0.09
{'5'}	{'V'}	0.1	{'K'}	{'B'}	0.11	{'Z'}	{'2'}	0.07
{'6'}	{'6'}	0.21	{'L'}	{'L'}	0.29			
{'7'}	{'P'}	0.08	{'M'}	{'M'}	0.36			
{'8'}	{'8'}	0.2	{'N'}	{'M'}	0.16			

Figura 3.2. Resultat d’analitzar la imatge *Joc_de_caracters_deformats II.png*

Com podem veure, al analitzar una imatge més distorsionada, el nostre classificador i les característiques que utilitza no són suficients per distingir totes els elements. Alguns s’identifiquen bé, però la majoria es confonen amb altres de semblants.

Finalment, he analitzat les imatges de dos matricules per aplicar el script una mica més al "mon real". Les dos matricules analitzades són les següents:

Figura 3.3. Imatges originals i binaritzades de les matricules

I els seus resultats son, respectivament:

Name	Label	Score
5	{'5'}	0.18
6	{'6'}	0.14
4	{'4'}	0.32
9	{'9'}	0.26
J	{'J'}	0.43
S	{'S'}	0.23
N	{'G'}	0.2

Name	Label	Score
2	{'2'}	0.28
0	{'0'}	0.25
9	{'9'}	0.3
3	{'3'}	0.19
G	{'G'}	0.22
S	{'S'}	0.22
W	{'W'}	0.46

Figura 3.4. Resultats d'analitzar les dos matricules

Amb aquests podem veure que, al no estar gaire distorsionades, el script llegeix les matricules amb prou precisió. Dit això, encara comet algun error i tampoc obté puntuacions gaire altes pels caràcters que reconeix bé (la majoria es troben entre 0.2 - 0.3)

4. Annexos / Scripts

A continuació afegeixo els dos scripts utilitzats per aquesta sessió:

```
clear
close all

%----------------------------------------- SET UP THE CLASSIFIER ------------------------

I = rgb2gray(imread('Joc_de_caracters.jpg'));
BI = I < 128;
imshow(BI)
CC = bwconncomp(BI);

props =
regionprops(CC,'MajoraxisLength','MinoraxisLength','Solidity','Extent','Circularity','Eccen
tricity','Centroid','BoundingBox');

ratio = [props(:).MinorAxisLength]./[props(:).MajorAxisLength];
centroids = cat(1, props.Centroid);
boxes = cat(1,props.BoundingBox);
centroid_x1 = ([centroids(:,1)]'-[boxes(:,1)]');
centroid_y1 = ([centroids(:,2)]'-[boxes(:,2)]');
rel_cent_x1 = centroid_x1./[boxes(:,3)]';
rel_cent_y1 = centroid_y1./[boxes(:,4)]';

X = [ratio; props.Solidity; props.Extent; props.Eccentricity; props.Circularity;
rel_cent_x1; rel_cent_y1]';

OUT = {'0' '1' '2' '3' '4' '5' '6' '7' '8' '9' 'B' 'C' 'D' 'F' 'G' 'H' 'J' 'K' 'L' 'M' 'N'
'P' 'R' 'S' 'T' 'V' 'W' 'X' 'Y' 'Z'};

Classifier = TreeBagger(100, X, OUT);

%----------------------------------------- TEST WITH NEW IMAGES -------------------------

I = rgb2gray(imread('Joc_de_caracters_deformats.jpg'));
BI = I < 128;
BI = bwmorph(BI,'majority');
figure;
imshow(BI);
CC = bwconncomp(BI);

props =
regionprops(CC,'MajoraxisLength','MinoraxisLength','Solidity','Extent','Circularity','Eccen
tricity','Centroid','BoundingBox');

ratio = [props(:).MinorAxisLength]./[props(:).MajorAxisLength];
centroids = cat(1, props.Centroid);
boxes = cat(1,props.BoundingBox);
centroid_x = ([centroids(:,1)]'-[boxes(:,1)]');
centroid_y = ([centroids(:,2)]'-[boxes(:,2)]');
rel_cent_x = centroid_x./[boxes(:,3)]';
rel_cent_y = centroid_y./[boxes(:,4)]';

X = [ratio; props.Solidity; props.Extent; props.Eccentricity; props.Circularity;
rel_cent_x; rel_cent_y]';

[label,score] = predict(Classifier, X);
table(Classifier.ClassNames, label, max(score)', 'VariableNames',{'Name','Label','Score'})
```

Figura 4.1. Script utilitzat per analitzar les imatges enunciat

```
clear
close all

%---------------------------------------- SET UP THE CLASSIFIER ----------------------
-------------------------------------

I = rgb2gray(imread('Joc_de_caracters.jpg'));
BI = I < 128;
imshow(BI)

CC = bwconncomp(BI);

props =
regionprops(CC,'Area','MajoraxisLength','MinoraxisLength','Solidity','Extent','Circularity'
,'Eccentricity','EulerNumber','Centroid','BoundingBox');
ratio = [props(:).MinorAxisLength]./[props(:).MajorAxisLength];
boxes = cat(1,props.BoundingBox);
centroids = cat(1, props.Centroid);
centroid_x1 = ([centroids(:,1)]'-[boxes(:,1)]');
centroid_y1 = ([centroids(:,2)]'-[boxes(:,2)]');
rel_cent_x1 = centroid_x1./[boxes(:,3)]';
rel_cent_y1 = centroid_y1./[boxes(:,4)]';
% X = [props.Area; ratio_axis]';
X = [ratio; props.Extent; props.Circularity; props.Eccentricity; rel_cent_x1;
rel_cent_y1]';

OUT = {'0' '1' '2' '3' '4' '5' '6' '7' '8' '9' 'B' 'C' 'D' 'F' 'G' 'H' 'J' 'K' 'L' 'M' 'N'
'P' 'R' 'S' 'T' 'V' 'W' 'X' 'Y' 'Z'};

Classifier = TreeBagger(100, X, OUT);

%---------------------------------------- TEST WITH NEW IMAGES -----------------------
-------------------------------------

img = rgb2gray(imread('Matricula2093gsw.png'));
figure;
imshow(img);
BI = img < 65;
BI = bwmorph(BI,'majority');
figure;
imshow(BI);

CC = bwconncomp(BI);
props = regionprops(CC,'Area', 'MajoraxisLength',
'MinoraxisLength','Solidity','Extent','Circularity','Eccentricity','EulerNumber','Centroid'
,'BoundingBox');
ratio = [props(:).MinorAxisLength]./[props(:).MajorAxisLength];
boxes = cat(1,props.BoundingBox);
centroids = cat(1, props.Centroid);
centroid_x = ([centroids(:,1)]'-[boxes(:,1)]');
centroid_y = ([centroids(:,2)]'-[boxes(:,2)]');
rel_cent_x = centroid_x./[boxes(:,3)]';
rel_cent_y = centroid_y./[boxes(:,4)]';
% X = [props.Area; ratio_axis]';
X = [ratio; props.Extent; props.Circularity; props.Eccentricity; rel_cent_x; rel_cent_y]';

%matricula = ['5';'6';'4';'9';'J';'S';'N'];
matricula = ['2';'0';'9';'3';'G';'S';'W'];
X_mat = [X(:,:); X(:,:); X(:,:); X(:,:); X(1:2,:)];

[label,score] = predict(Classifier, X_mat);
label1 = label(1:7);
score1 = score(1:7,:)';
m = max(score1)';
table(matricula, label1, m, 'VariableNames',{'Name','Label','Score'})
```

Figura 4.2. Script utilitzat per analitzar matricules

5. Documentació

- https://es.mathworks.com/help/images/ref/regionprops.html