LAB PIV

Òscar Medrano Sánchez

Juny 2024

Sistema de Recuperació d'Imatges Basat en Descriptors de l'Estructura del Color

1 Introducció

Basant-se en la versió anterior, aquest informe presenta els avenços fets en el sistema de Recuperació d'Imatges Basada en Contingut (CBIR) realitzats al laboratori PIV. És important destacar que l'avaluació de la segona versió del sistema CBIR continua emprant la mateixa base de dades de 2.000 imatges amb 500 objectes diferents. Això assegura la consistència i permet una comparació directa entre les dues versions.

Basant-se en la primera versió, la segona iteració incorpora diverses modificacions i introdueix la implementació del descriptor de color. Integrant l'espai de color HMMD (Hue-Max-Min-Diff) de l'estàndard MPEG-7, millorem la capacitat del sistema per caracteritzar les imatges basant-se en la seva informació de color. Aquesta addició complementa l'enfocament basat en histogrames utilitzat anteriorment, permetent una anàlisi més completa del contingut de les imatges.

2 Descripció global del sistema

La descripció general del sistema per a la segona versió es manté igual que en la primera versió, amb la diferència en els descriptors utilitzats en el sistema. Enlloc de representar les imatges com a histogrames, la segona versió utilitza l'HMMD, que també es representa com un vector denotat com h1.

El sistema pren un fitxer d'entrada, "input.txt", que conté els noms de les imatges a ser analitzades. Per a cada imatge especificada en el fitxer d'entrada, el sistema realitza els següents passos:

1. Carrega la matriu H i extreu el vector HMMD, denotat com h1, de la imatge actual.
2. Calcula el vector de distàncies, d, entre el vector HMMD h1 i tots els vectors HMMD en H.

3. Identifica els 10 valors més petits en el vector de distàncies d i reté els seus índexs corresponents per determinar els 10 noms d'imatges a escriure en el fitxer de sortida, "output.txt".

Algorithm 1: System

Input: Input file *input.txt*
Output: Output file *output.txt*
load('dataBase');
T = createTable();
for $i \leftarrow 1$ **to** $NumImages$ **do**
    im = rgb2dsh(input(j), T);
    h1 = imhist(readImage(im));
    for $j \leftarrow 1$ **to** 2000 **do**
        d(j) = distanceX(h1, H(j,:))
    sort(d);
    for $j \leftarrow 1$ **to** 10 **do**
        write(Output, d(j));
return Output

En la funció *'rgb2dsh()'*, em refereixo a 'dsh' per Diferencia, Suma i Hue, explicats a continuació.

Espai de color: HMMD

En la segona versió del sistema CBIR, vam passar d'utilitzar imatges en blanc i negre a imatges RGB, cosa que ajuda a captar la informació del color.

L'espai de color HMMD consisteix en quatre components de color anomenats H-tonalitat (hue), M-màxim (maximum), M-mínim (minimum) i D-diferència (difference). Els components de l'espai de color HMMD es deriven dels valors RGB. Aquesta conversió es fa de la següent manera:

$$Maxim = max(R, G, B) \tag{1}$$

$$Minim = min(R, G, B) \tag{2}$$

$$Diferencia = Maxim - Minim \tag{3}$$

$$Suma = \frac{Maxim + Minim}{2} \tag{4}$$

El Hue és una mica més complex i farragós, per això la conversió la faig utilitzant una funció que converteix la imatge de RGB a HSV, i em guardo directament la matriu d'H (corresponent a Hue).

Algorithm 2: getDSH

Input: Input image *imag*

```
R = imag(:, :, 1);
G = imag(:, :, 2);
B = imag(:, :, 3);
S = (max(max(R, G), B) + min(min(R, G), B)) / 2;
D = max(max(R, G), B) - min(min(R, G), B);
imag = rgb2hsv(imag);
H = floor((255/32)*imag(:,:,1)); //for 128 bins
```

Faig èmfasi que aquesta és la manera de calcular-ho pel cas de 128 bins, ja que necessito que H tingui valors entre 0 i 7, mentre que si fos pel cas de 256 bins, necessitaríem de 0 a 15; degut al Hue màxim en cada un d'ells, com s'il·lustra en la Figura 1. De fet, durant el codi hi ha parts que estan comentades, això és degut a que depèn en quin moment estem, si en 128 o 256 bins, hi ha funcions que s'han de comentar i funcions que descomentar, ja que algunes són útils per un cas i les altres per l'altre.

El següent pas és quantificar els quatre nivells de l'espai de color HMMD condensant-los en un únic nivell mitjançant la taula següent. Utilitzant la taula proporcionada, podem assignar un nivell únic a cadascun dels components de color HMMD, cosa que ens permet representar quantitativament l'espai de color HMMD i realitzar càlculs o anàlisis posteriors basats en aquest. En el cas del nostre sistema, hi ha la possibilitat de quantificar la imatge de dues maneres: 128 nivells i 256 nivells. En el meu sistema he fet aquesta quantificació en una funció dita 'CreateTable()' (per 128 bins) o 'CreateTable256()' (per 256 bins), ja que tracto la transformació com una taula amb 3 entrades (DSH) i una sortida.

No. of cells	256		128	
Subspace	Hue	Sum	Hue	Sum
0	1	32	1	16
1	4	8	4	4
2	16	4	8	4
3	16	4	8	4
4	16	4	8	4

Figure 1: Quantificació HMMD

Un cop creada la taula de quantificació, el següent pas és mapejar els valors de les matrius

D, S i H de cada una de les imatges a la quantificació pertinent. És fa de la següent manera:

Algorithm 3: Quantification image

Output: Output image $NovaIm$
for $i \leftarrow 1$ **to** 480 **do**
  for $j \leftarrow 1$ **to** 640 **do**
    NovaIm(i, j) = Table128(D(i,j)+1, S(i,j)+1, H(i,j)+1);

La *'Table128'* és la taula creada per *'CreateTable()'*.

De fet, els algorismes 2 i 3 és fan junts dins la funció *'rgb2dsh()'*.

Després de quantificar la imatge, el procés que quedava és el mateix que a la primera part de la pràctica, utilitzant però, més distàncies. Aquestes són les que he incorporat:

MAD

La distància de desviació mitja absoluta està definida com:

$$d(h1, h2) = \sum_{n=0}^{N-1} |h1 - h2| \tag{5}$$

Sent *h1* i *h2* els histogrames que li entrem per funció, i així serà a totes les definicions de distàncies.

MSE

La distància de l'error quadràtic mitjà està definida com:

$$d(h1, h2) = \sum_{n=0}^{N-1} |h1 - h2|^2 \tag{6}$$

Cosinus

La distància de Cosinus està definida com:

$$d(h1, h2) = 1 - \frac{\Sigma h1 \cdot h2}{\sqrt{\Sigma \text{hist}}} \tag{7}$$

Divergencia de Kullback-Leibler

La distància de la Divergencia de Kullback-Leibler està definida com:

$$d(h1, h2) = \sum_{n=0}^{N} h1 \log\left(\frac{h1}{h2}\right) \tag{8}$$

3 Resultats

3.1 128 bins

Primerament, avaluarem les distàncies per quina ens proporciona més F-score, calculat a partir de la Precision i del Recall (programa anomenat *'precisionrecall.m'*). Aquest és el resultat:

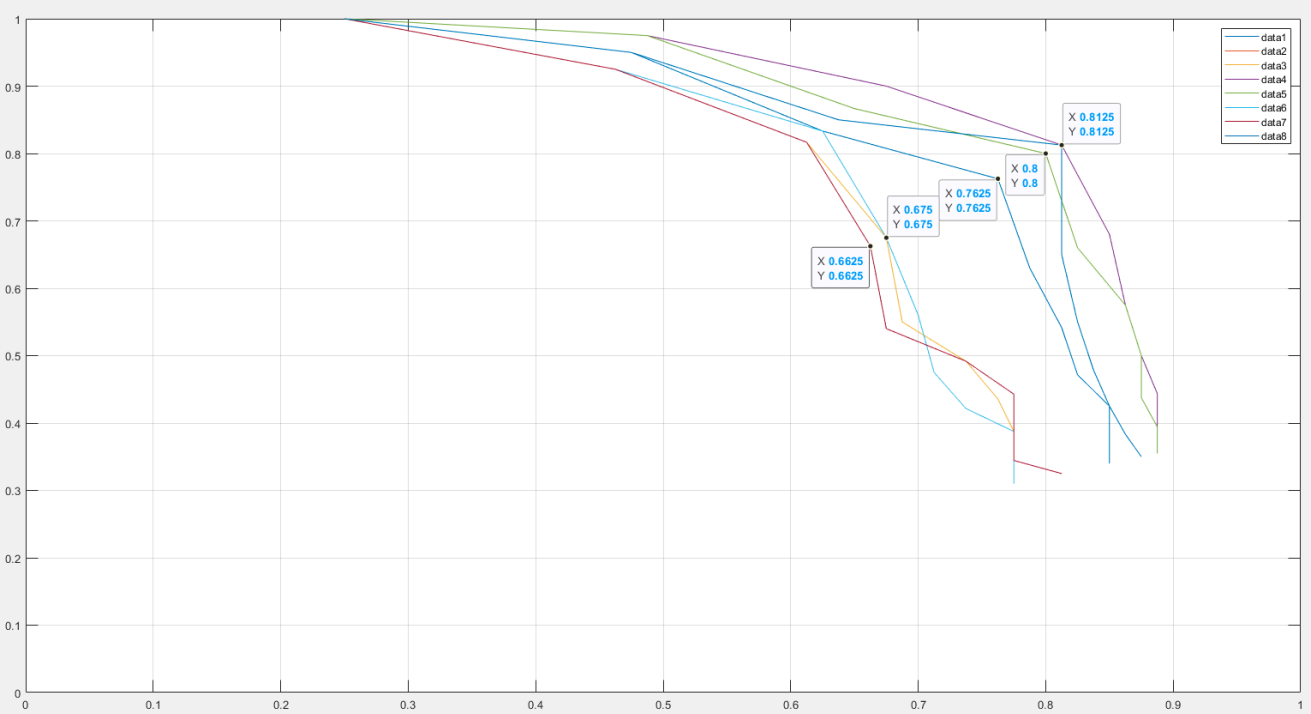

Figure 2: Taula d'F-score per 128 bins

Tot i fer-ho per 8 distàncies diferents, només veiem 7 gràfiques. Això és degut a que amb la distància de Bhattacharya (data 2) i Hellinger (data4) s'aconsegueix el mateix resultat, que és precisament la gràfica que va més amunt i el resultat més alt, 0.8125. També destacar la divergència de Kullback-Leibler (data8), que aconsegueix el mateix F-score que les dues distàncies anteriors.

L'ordre de la llegenda és el següent, de primer a última: MAD, Bhattacharyya, Correlació, Hellinger, Txi-Quadrat, MSE, Cosinus i Divergencia de Kullback-Leibler.

Un cop vist els F-score, entre els que millors funcionen mirem quin té menys cost computacional.

Distància	**F-score**	**Temps (s)**
Bhattacharya	0.8125	0.470332
Hellinger	0.8125	0.497630
Divergència de Kullback-Leibler	0.8125	0.560998

Table 1: Comparativa de distàncies pel cost computacional per 128 bins

El temps està calculat en temps total que tarda tot el programa a decidir tot el fitxer d'output. Com podem observar, la més eficient és la distància de Bhattacharya, seguit de Hellinger i, per acabar, la Divergència de Kullback-Leibler. Cal recalcar que la diferència de valors absoluts era la més ràpida de totes, però al no ser la millor en F-score, ens quedarem com ha millor distància pel cas de 128 bins amb la distància de **Bhattacharya**.

El cost computacional referent és el de l'ordinador utilitzar en el laboratori, amb el següents temps:

Computer Type	LU	FFT
Windows 10, AMD Ryzen Threadripper(TM) 3970x @ 3.50 GHz	0.1930	0.1892
Debian 9(R), AMD Ryzen Threadripper 3970x @ 3.50 GHz	0.2612	0.1259
This machine	0.4694	0.3388

Figure 3: Cost computacional de referència

L'operació que s'assembla més a la nostra és la FFT, per tant ens quedem amb **0.3388** segons com a temps referent. Cal recalcar que absolutament tots els càlculs de temps d'aquesta memòria s'han fet amb el mateix ordinador que en el calculat els temps referent.

3.2 256 bins

Com hem fet anteriorment, primer avaluarem les distàncies per quins ens proporciona més F-score, calculat a partir de la Precision i del Recall. Aquest és el resultat:

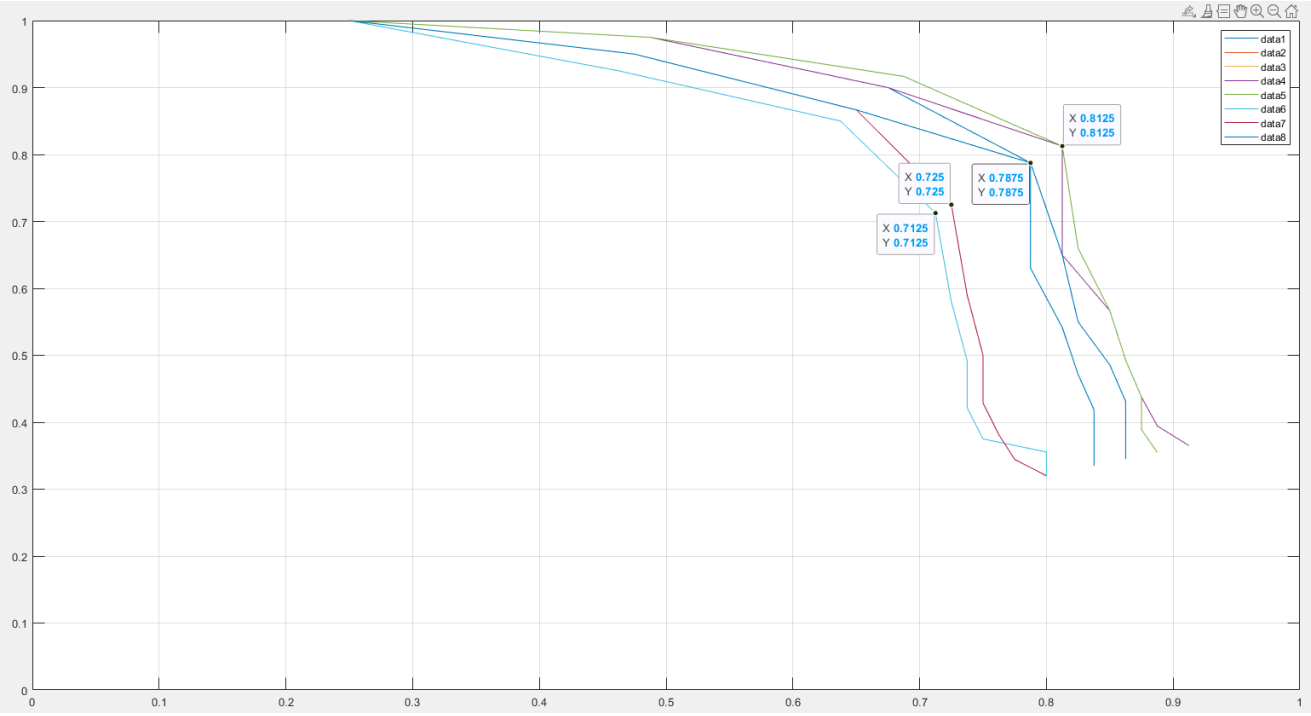

Figure 4: Taula d'F-score per 256 bins

Passa una cosa semblant a l'anterior. Són tres les distàncies que aconsegueixen un F-score de 0.8125, dos de les quals tenen exactament la mateixa gràfica, que són la de Bhattacharya (data2) i la de Txi-Quadrat (data 5); i una tercera que és la distància de Hellinger (data 4).

Un cop vist els F-score, entre els que millors funcionen mirem quin té menys cost computacional.

Distància	**F-score**	**Temps (s)**
Bhattacharya	0.8125	0.495955
Hellinger	0.8125	0.537909
Txi-Quadrat	0.8125	0.458780

Table 2: Comparativa de distàncies pel cost computacional per 256 bins

De la mateixa manera que abans, el temps està calculat en temps total. Com poder observar, la distància més eficient és la distància de Txi-Quadrat. El programa sencer tarda 4 centèssimes menys que la distància de Bhattacharya i 8 que Hellinger. Per tant, em quedo com a millor distància pel cas de 256 bins amb la distància de **Txi-Quadrat**.

Mencionar també, que l'ordre de distàncies imprimides a la gràfica és el mateix ordre que per la gràfica de 128 bins.

4 Conclusions

Sistema	Descriptor	Bims	Dist	F	Temps/im	T (fftlab)
1	Histograma NGris	128	Bhattacharya	0.5	13 ms	0.3388 s
2	HMMD	128	Bhattacharya	0.8125	23.5 ms	0.3388 s
3	HMMD	256	Txi-Quadrat	0.8125	22.9 ms	0.3388 s

Table 3: Comparativa

Cal destacar que els resultats presentats aquí es basen en un experiment específic, concretament les imatges de l'input.txt, i poden no ser generalitzables a altres contextos. Per tant, els resultats s'han d'interpretar amb precaució i pot ser necessària més experimentació amb altres input.txt per confirmar les conclusions.

El programa anterior utilitzava un histograma de nivells de gris, centrant-se únicament en la intensitat de llum, això implica perdre molta informació rellevant de la imatge. Això volia dir, que si hi havia canvis de llum, el programa no era capaç de detectar-los. El nou programa en canvi, a part de tenir en compte la luminància, com ja feia l'anterior, també té en compte el color. Amb aquesta nova informació, s'ha guanyat molta més informació sobre la imatge. Per això mateix, els resultats obtinguts s'alineen amb les expectatives de millora.

En quant a les millors distàncies, com ja va passar a la primera part del projecte, tant les distàncies de Bhattacharya, com de Hellinger són les que millors resultats donen. Em sorprèn què a elles se li afegeixi la de Txi-Quadrat, distància que a la primera part no va donar tants bons resultats. A més, el valor absolut de la diferència dona resultats sorprenentment fiables, amb un F-score superior al 0.7.

A més, m'ha sorprès gratament una nova distància que he afegit en aquesta segona part, la divergència de Kullback-Leibler, on amb 128 bins ha arribat a igualar als millors resultats possibles aconseguits per qualsevol de les distàncies provades.

Tanmateix, la configuració òptima per al sistema es basa en un experiment específic i pot no ser generalitzable a altres contextos. Pot ser necessària més experimentació amb altres fitxers d'entrada per confirmar les conclusions.

Millores

Podem millorar el sistema de detecció d'imatges en dues àrees principals: la descripció de les imatges i el procés de presa de decisions. Per a la descripció de les imatges, podríem explorar tècniques alternatives, com ara l'ús de descriptors basats en textures, com el Local Binary Patterns (LBP), que poden oferir una caracterització més detallada i robusta de les estructures dins de les imatges. En la fase de presa de decisions, una millora potencial seria implementar un sistema d'aprenentatge automàtic com a decisor, entrenant-lo amb els descriptors obtinguts per tal que el sistema aprengui a prendre decisions més precises i informades.

Un altre enfocament innovador seria substituir completament el sistema actual de descriptors i decisió per una solució basada en aprenentatge profund. Utilitzant una xarxa neuronal convolucional (CNN) per gestionar totes dues etapes de manera consecutiva.