Pràctica 3: Classificació de Textures amb Matrius de Coocurrència

Lluís F Collell i Miguel Agundez

Breu introducció:

La textura és una característica fonamental en el processament i anàlisi d'imatges, especialment útil en tasques com la segmentació i classificació d'imatges. En aquesta pràctica es treballa amb descriptors de textura obtinguts mitjançant matrius de coocurrència, una tècnica estadística senzilla i eficient que captura la relació espacial entre les intensitats de píxel dins una imatge.

L'objectiu principal de la pràctica és extreure descriptors de textura per a cada imatge i utilitzar-los per classificar-les en 28 classes diferents. Es treballarà amb una base de dades que conté 80 imatges per classe, utilitzant-ne 40 per entrenament i 40 per test. El procés de classificació es realitzarà mitjançant el classificador dels k-veïns més propers (k-NN).

Mitjançant aquesta pràctica, l'estudiant adquirirà una comprensió pràctica de l'ús de descriptors de textura per a la classificació automàtica d'imatges, així com una introducció als mètodes estadístics aplicats en visió per computador.

Índex de continguts:

1. Introducció

La **textura** és una característica visual clau en el **processament d'imatges**. Permet descriure patrons repetitius o variacions de tons que no es poden captar amb el color o la forma. És especialment útil en aplicacions com:

- **Inspecció de superfícies**
- **Diagnòstic mèdic**
- **Reconeixement d'objectes**

En aquesta pràctica es treballa amb una **base de dades de textures** que conté **28 classes diferents**, amb **40 imatges per entrenament** i **40 per test** a cada classe.

L'objectiu és:

- **Descriure la textura** de cada imatge mitjançant **descriptors estadístics**
- **Classificar les imatges** utilitzant el classificador **k-Nearest Neighbours (k-NN)**
- **Avaluar el rendiment** amb la **matriu de confusió** i el percentatge d'encerts (**accuracy**)

Per descriure la textura, utilitzarem les anomenades **matrius de coocurrència (GLCM)**, que tenen en compte la relació entre píxels propers.

2. Fonaments teòrics i mètode

2.1 Textura

La **textura** és una propietat visual que descriu l'estructura d'una superfície. Es basa en la **distribució espacial dels valors de gris** en una imatge i pot expressar característiques com la **rugositat**, **regularitat** o **granularitat**. Per descriure una textura, cal observar una regió i no un sol píxel.

2.2 Matrius de coocurrència (GLCM)

Les **matrius de coocurrència** compten quantes vegades apareixen junts dos valors de gris a una certa **distància** i **direcció** dins la imatge. Aquestes relacions entre píxels permeten capturar patrons de textura.

Paràmetres principals:

- **Distància** entre píxels (per exemple: 1, 3, 5...)
- **Angle** (0°, 45°, 90°, 135°)
- **Nombre de nivells de grisos** (es pot reduir per eficiència)

2.3 Estadístics

A partir de la GLCM s'extreuen descriptors que resumeixen les propietats de la textura:

- **Contrast**: mesura la variació local d'intensitat
- **Energia**: indica repetició o uniformitat
- **Homogeneïtat**: valors similars a prop
- **Entropia**: grau de desordre o complexitat

> Nota: existeixen altres estadístics com la **correlació**, la **variància** o la **inversió del moment diferencial**, que no s'han utilitzat en aquesta pràctica.

2.4 Classificador k-NN

El classificador **k-Nearest Neighbours** assigna una classe a una imatge basant-se en les **k imatges més properes** del conjunt d'entrenament. És senzill, no requereix entrenament complex i funciona bé amb dades representatives.

2.5 Justificació del valor de *k*

S’utilitza **k = 1**, el valor mínim vàlid, perquè:

- **k < 1 no és vàlid**, ja que no es pot considerar menys d’un veí
- Valors majors poden disminuir l’accuracy si hi ha solapament entre classes

Amb **k = 1**, cada imatge es classifica segons el seu **veí més proper**, un enfocament simple però efectiu en molts casos.

3. Fonaments tècnics i implementació

3.1 Càrrega i processament de dades

Per iniciar la pràctica, es carreguen les imatges de 28 carpetes corresponents a cada classe de textura. Cada carpeta conté 80 imatges:

- Les primeres 40 s'utilitzen com a conjunt d'entrenament.
- Les altres 40 s'utilitzen per al test.

El bucle principal recorre cada carpeta i crida la funció *'computeFeatureVector'* per extreure el vector de característiques de cada imatge. Les característiques de les imatges d'entrenament es guarden a la variable *'vecTrain'*, i les del test a *'vecTest'*. Les etiquetes corresponents es guarden a *'labTrain'* i *'labTest'*.

3.2 Extracció de característiques (computeFeatureVector)

La funció *'computeFeatureVector'* és la responsable de transformar una imatge en un vector numèric que descriu la seva textura. El procés inclou:

- Conversió a escala de grisos si la imatge és RGB.
- Assegurar que la imatge està en format 'uint8'.
- Generació de matrius de coocurrència amb múltiples desplaçaments.
- Extracció dels estadístics de textura: contrast, energia, homogeneïtat i correlació.
- Càlcul de valors estadístics bàsics: mitjana i desviació estàndard de la imatge.
- Reducció i normalització d'un histograma a 8 bins.

Tots aquests valors es concatenen en un únic vector, que representa la textura de la imatge d'una manera compacta.

3.3 Configuració dels offsets i distàncies

Els *offsets* defineixen la posició relativa entre píxels utilitzada per generar les matrius de coocurrència. En aquest cas, es treballa amb 4 direccions:

- (0, d) → horitzontal
- (–d, d) → diagonal superior dreta

- (–d, 0) → vertical
- (–d, –d) → diagonal superior esquerra

Les distàncies seleccionades són d = 3 fins a 6, permetent capturar patrons de textura a diferents escales.

3.4 Ús de funcions de MATLAB: graycomatrix, graycoprops, fitcknn, predict

- *'graycomatrix'*: genera les matrius de coocurrència (GLCM) a partir de la imatge, segons els offsets indicats. Inclou l'opció 'Symmetric' per considerar la matriu simètrica.
- *'graycoprops'*: calcula estadístics de textura a partir de la GLCM, com contrast, energia, homogeneïtat i correlació.
- *'fitcknn'*: entrena un model de classificació k-NN utilitzant els vectors d'entrenament. En aquest cas, s'ha utilitzat un valor de k = 1 i s'han normalitzat les característiques.
- *'predict'*: aplica el model entrenat per predir la classe de cada imatge del conjunt de test.

3.5 Fragments de codi rellevants

- Assignació d'offsets:
    - Per cada valor de d entre 3 i 6, es generen 4 direccions que es concatenen en un array.
- Crida a la funció *'graycomatrix'* amb els offsets generats per construir les GLCM.
- Extracció dels estadístics amb *'graycoprops'* i càlcul de valors complementaris com la mitjana i la desviació.
- Entrenament del classificador amb *'fitcknn'* i predicció amb *'predict'*.
- Avaluació del resultat amb *'confusionmat'* i càlcul de l'accuracy.

4. Resultats i anàlisi

4.1 Descripció del conjunt de dades

El conjunt de dades utilitzat conté un total de 28 classes de textures, organitzades en carpetes. Cada classe representa una textura diferent amb múltiples mostres.
Les imatges estan en format '.jpg' i tenen una mida i qualitat adequades per l'anàlisi de textura.

4.2 Accuracy i matriu de confusió

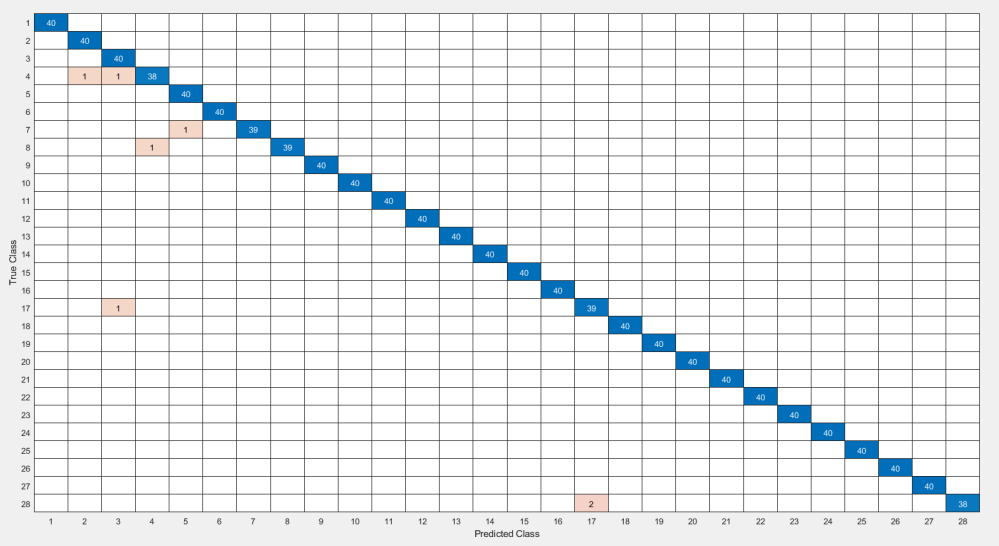
El rendiment del sistema de classificació es mesura mitjançant:

- La matriu de confusió, que mostra, de manera gràfica, la correspondència entre les classes reals i les classes predites.

- L'accuracy, que representa el percentatge d'imatges classificades correctament.

El càlcul de l'accuracy es fa amb la fórmula:

Accuracy = (suma de la diagonal de la matriu de confusió) / (total d'elements)



4.3 Impacte dels paràmetres: k, distància, angles

S'han analitzat diversos paràmetres que poden influir en el rendiment del sistema:

- Valor de k: s'ha fet servir k = 1 per simplicitat i perquè sol oferir bons resultats quan les dades estan ben separades. Valors superiors poden disminuir la sensibilitat a valors atípics però també poden reduir l'accuracy.

- Distància entre píxels (d): s'han provat distàncies de 3 a 6. Distàncies més grans poden captar estructures més globals, però poden fer perdre detall.

  Aquest llindar entre 3 i 6 l'hem establit a mesura d'anar experimetnar amb el nsotre codi observant el resultat de l'acucracy per cada recorregut del codi. Tot i això, valors més petits de 3 o més grans de 6 donen resultats molt similars.

- Angles (offsets): utilitzar múltiples direccions (0°, 45°, 90°, 135°) millora la robustesa del descriptor, ja que considera patrons amb orientacions diferents.

  Ara bé per augmetnar el nomrbe de matriu de coocurrències i obtenir un accuracy amb mñes precisió afegim a la funió graycomatrix l'expressió ("Symmetric", true) que utilitzarà els valors sim+ètrics als angles mencionats prèvimanet (180°, 225°, 270°, 315°).

Aquest conjunt de paràmetres s'ha seleccionat buscant un equilibri entre rendiment i temps de còmput.

Avaluació paràmetres:

INFLEXIÓ NUMERO DE CARACTERÍSTIQUES APLICADES

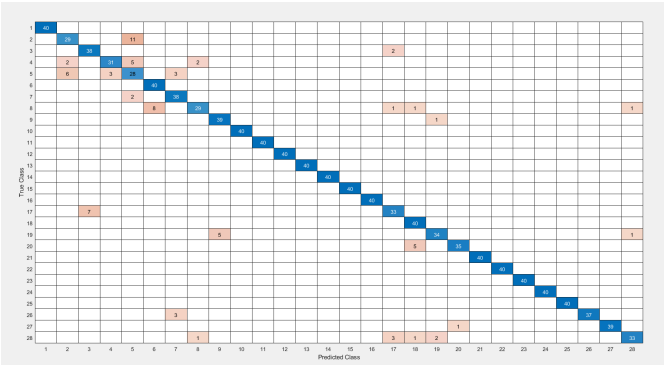
Fixarem un valor de K i uns valors d'Offset i anirem avaluant els resultats de 'accuracy' segons el numero de característiques aplicades.

1 característica : 4 característiques : 7 característiques.

Valors fixats

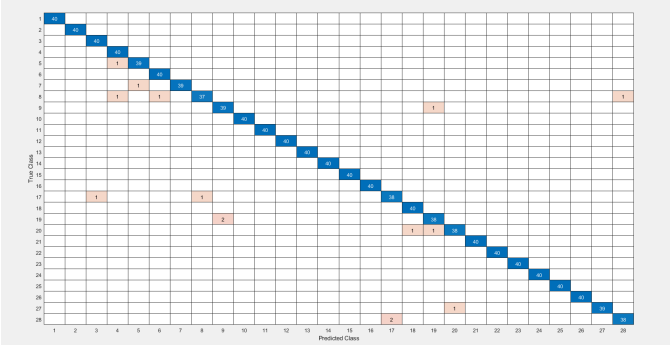
- K=1; OFFSETS 3:6

Una única característica: 'Constrast'.
Obtenim el següent resultat:



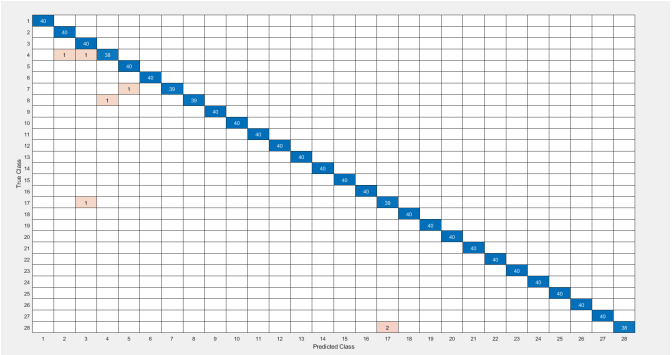
Accuracy calculat = 0.9313

Quatre característiques: ‘Constrast’, ‘Energy’, ‘Homogeneity’ i ‘Correlation’.
Obtenim el següent resultat:



Accuracy trobat = 0.9866

Set característiques: ‘Constrast’, ‘Energy’, ‘Homogeneity’, ‘Correlation’, ‘Mean’, ‘Std’ i ‘Histograma’.
Obtenim el següent resultat:



Accuracy trobat = 0.9938

CONCLUSIÓ

Ens trobem que, tal i com esperàvem, a l’afegir més característiques al model obtenim un valor d’accuracy més elevat.
Ja que el nombre de matrius de coocurrències és major, per tant, la informació obtinguda és proporcional a aquest increment.

INFLEXIÓ K

Fixarem 7 característiques i anirem canviant els valors ‘K’ i ‘Offset’

Valors fixats:

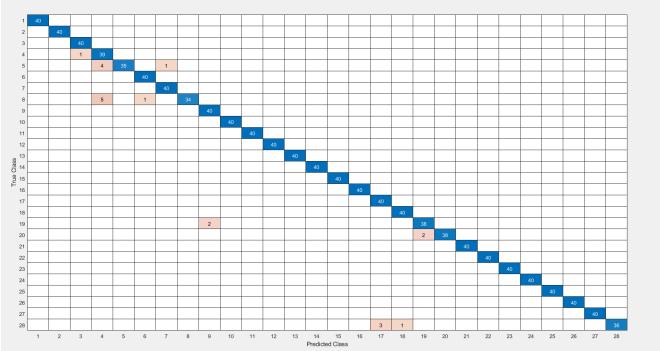
- Set característiques: ‘Constrast’, ‘Energy’, ‘Homogeneity’, ‘Correlation’, ‘Mean’, ‘Std’ i ‘Histograma’.
- OFFSETS 3:6

K=1

Imatge anterior

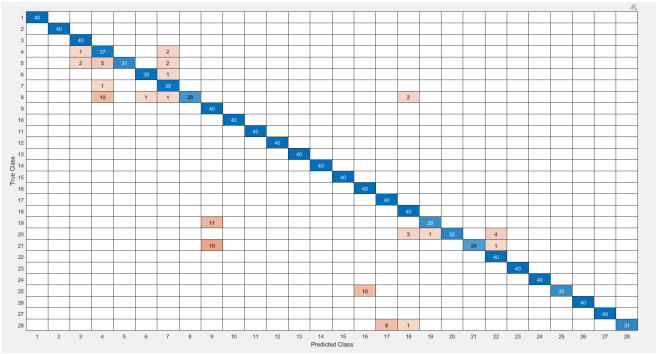
Accuracy trobat = 0.9938

K=10



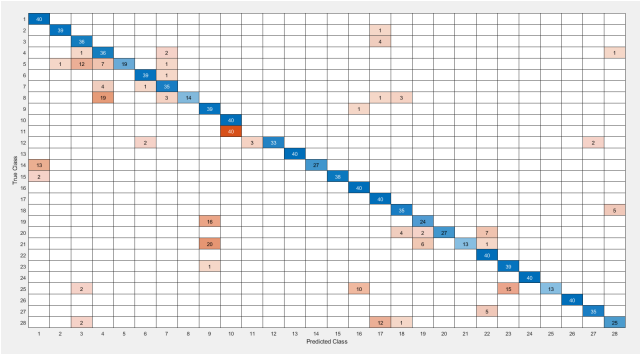
Accuracy trobat = 0.9821

K=50



Accuracy trobat = 0.9268

K=100



Accuracy trobat = 0.7911

CONCLUSIÓ

A mesura que augmentem el valor de **K** en el classificador **K-NN**, observem una davallada progressiva en l'**accuracy** del sistema.

Aquest comportament es deu al fet que, en considerar un nombre més elevat de veïns, el classificador tendeix a **suavitzar massa les fronteres entre classes**, fet que pot provocar que exemples amb veïns majoritàriament d'altres categories siguin classificats incorrectament.

En altres paraules, quan **K és petit (per exemple, 1)**, la classificació es basa en el veí més proper, capturant millor els detalls locals de la distribució. En canvi, amb valors **molt grans de K**, el model es fa més insensible a les diferències locals i pot acabar **generalitzant en excés**, cosa que redueix la precisió en casos amb classes poc equilibrades

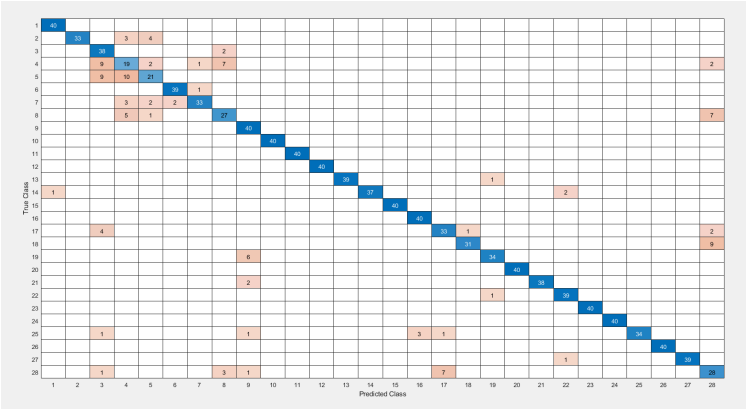
INFLEXIÓ OFFSETS

Fixarem 7 característiques i el valor K, per anar canviant la llargària de la distància dels Offsets

Valors fixats:

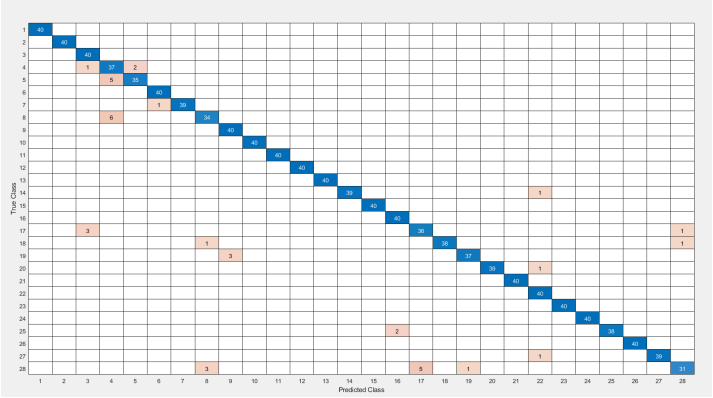
- Set característiques: 'Constrast', 'Energy', 'Homogeneity', 'Correlation', 'Mean', 'Std' i 'Histograma'.
- K=1

Offsets 2:100



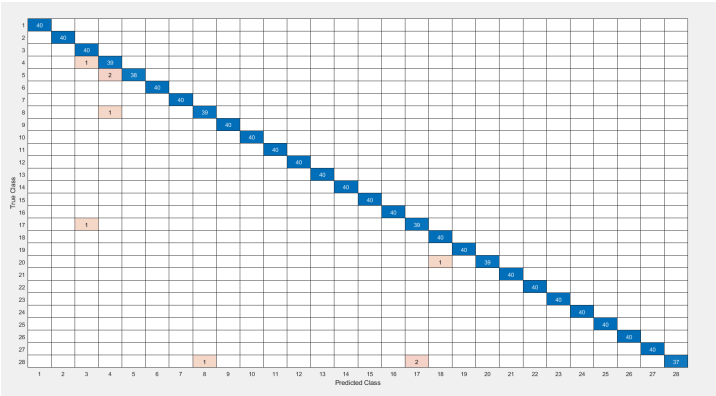
Accuracy trobat = 0.8946

Offsets 2:50



Accuracy trobat = 0.9661

Offsets 1:10



Accuracy trobat = 0.9920

CONCLUSIÓ

A mesura que s'incrementa el rang d'offsets (2:50 i 2:100), l'accuracy disminueix progressivament perquè s'introdueix informació menys rellevant i potencialment sorollosa al model.

4.4 Anàlisi dels errors per classes

A partir de la matriu de confusió, es pot observar que:

- Algunes classes presenten una accuracy molt alta, amb encerts gairebé del 100%.
- Altres tenen confusions recurrents amb classes similars visualment. Aquestes confusions solen aparèixer quan les textures comparteixen patrons, intensitats o orientacions.

Aquestes confusions poden indicar la necessitat de millorar la descripció (per exemple, afegint més estadístics o utilitzant filtres multiescala) o bé aplicar classificadors més avançats.

5. Conclusions

5.1 Resum del rendiment obtingut

El sistema de classificació de textures implementat ha assolit un **accuracy màxim del 99,38%**, evidenciant una gran precisió en el reconeixement de les diferents classes del conjunt de dades.

Aquest resultat s'ha obtingut combinant:

- Set característiques de textura: *contrast*, *energy*, *homogeneity*, *correlation*, *mean*, *std* i *histograma*
- Offsets per a distàncies de 3 a 6
- Un classificador *k-NN* amb *k = 1*

Les proves realitzades han permès observar que **a mesura que s'incrementa el nombre de característiques**, també ho fa l'accuracy del sistema, passant del 93,13% (amb només *contrast*) al 99,38% (amb les set característiques).

També s'ha pogut comprovar que l'ús de **diferents offsets i l'opció 'Symmetric'** millora la cobertura angular i, per tant, la capacitat descriptiva dels descriptors. Finalment, els valors petits de *k* ofereixen millors resultats que valors més elevats, evitant la sobre-generalització del model.

5.2 Limitacions detectades

Malgrat els bons resultats, s'han detectat algunes limitacions del sistema:

- Dificultat per discriminar textures amb patrons molt similars.
- L'ús extens de GLCM pot augmentar el temps de processament, sobretot amb molts offsets.
- El classificador k-NN no fa aprenentatge intern i pot veure afectat el seu rendiment per la distribució desigual de les dades.
- Offsets massa llargs poden aportar informació sorollosa i perjudicar l'accuracy.

5.3 Propostes de millora

A partir de l'anàlisi feta, es poden proposar diverses línies de millora per futures implementacions:

- **Normalització prèvia** de les imatges o les característiques per reduir l'impacte de valors extrems.

- **Exploració d'altres descriptors** com *Local Binary Patterns* o *filtres de Gabor* per enriquir la representació de textures.

- **Prova amb classificadors més avançats** com *SVM*, *Random Forest* o xarxes neuronals, per millorar la capacitat de separació.

- **Reducció de dimensionalitat** amb tècniques com *PCA* per simplificar l'espai de característiques i reduir temps de computació.

- **Validació creuada** per optimitzar els paràmetres com *k*, *número d'offsets* o *combinació de descriptors*.