Universitat Autònoma de Barcelona

Facultat de Ciències

ETIQUETADOR AUTOMÀTIC

Intel·ligència Artificial

Nora Alquézar Presta - 1671727

Curs 2023-24

1 Introducció

En aquest treball, abordem un problema d'etiquetatge automàtic d'imatges de roba utilitzant els algoritmes K-Means i KNN. L'objectiu és desenvolupar un sistema que pugui aprendre a etiquetar un seguit d'imatges de roba segons el tipus de peça i color.

Per simplificar el procés, hem seleccionat un conjunt reduït d'etiquetes que inclou 8 tipus de peces de roba i 11 colors bàsics.

En primer lloc, utilitzem l'algoritme K-Means per identificar els colors predominants de cada imatge, el qual és un mètode de classificació no supervisada que agrupa les dades en K clusters, basant-se en les seves similituds en termes d'atributs. A diferència d'aquest, l'algoritme KNN assigna etiquetes de peces de roba, en aquest cas és un algoritme de classificació supervisada que cerca els veïns més propers a una mostra d'entrada per assignar-li una etiqueta.

2 Mètodes d'anàlisis implementats

2.1 Funcions d'anàlisi qualitatiu

Les funcions d'anàlisi qualitatiu ens permeten avaluar la coherència i consistència de les etiquetes assignades als diferents conjunts d'imatges pels algorismes KNN i K-Means.

El que busquem és observar si les etiquetes assignades reflecteixen adequadament els atributs i característiques desitjats. Per exemple, en el cas de la funció de Retrieval_by_color amb el color, en el cas de Retrieval_by_shape amb la forma o bé amb la combinació dels dos amb Retrieval_combined.

A continuació, aprofundim en cadascuna de les funcions mencionades en el disseny experimental que s'ha empleat per a provar-les i els resultats obtinguts.

2.1.1 Retrieval by color

Aquesta funció busca les imatges que contenen el color desitjat. Per poder realitzar això, recorrem cada imatge i les seves etiquetes de color corresponents. Si totes les etiquetes del color desitjat es troben presents en les etiquetes de l'imatge, aquesta s'afegeix a la llista de resultats. A més, l'hem millorat afegint un paràmetre d'entrada que conté el percentatge del color desitjat present a cada imatge de la llista i fent que retorni les imatges ordenades pel percentatge. Per tal de fer això, hem creat una funció anomenada unique_colors_percentages, la qual ens retornarà una nova llista de colors, sense color repetits i una llista ordenada amb els percentatges del color especificat de cada imatge.

Per tal de provar aquesta funció, cal haver inicialitzat el K-Means, predit les etiquetes, els percentatges que ens interessen i per últim, cridar a la funció Retrieval_by_color. És per això que s'ha de declarar com a true el booleà retrieval_by_color.

Per comprovar l'eficàcia d'aquesta funció, li passem una cerca determinada i analitzem visualment les imatges filtrades amb els valors de colors predits, gràcies a la funció visualize_retrieval, ja que permet visualitzar aquestes imatges permetent veure més clarament la peça de roba que se'ns mostra.

Provem Retrieval_by_color utilitzant els paràmetres:
Retrieval_by_color(imgs, best_color_labels, best_color_percentages, colors, 8), on colors = ["Red"] i 8 el nombre d'imatges que volem visualitzar. I obtenim:

Figure 1: Red

I també utilitzant els paràmetres:
Retrieval_by_color(imgs, best_color_labels, best_color_percentages, colors, 8), on colors = [“Black”] i 8 el nombre d’imatges que volem visualitzar. I obtenim:

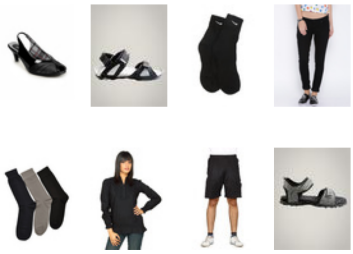

Figure 2: Black

2.1.2 Retrieval by shape

Pel que fa a la funció Retrieval_by_shape utilitzarem el mateix mètode que la funció anterior (Retrieval_by_color).Començarem amb la predicció de classes i percentatges. A continuació, cridarem a la funció Retrieval_by_shape tot declarant a true el booleà Retrieval_by_shape que fa possible la seva execució, com també la predicció i percentatges anteriors. En aquest cas, per realitzar la comparació visual d’imatges, en lloc de fer-ho a partir dels colors predits, ho farem a partir de la forma.

Realitzem la prova utilitzant:
Retrieval_by_shape(test_images, best_color_labels, best_color_percentages, shapes, 8), on shapes és [”Shirts”] i 8 el nombre d’imatges que volem visualitzar. I obtenim:

Figure 3: Shirts

I utilitzant: Retrieval_by_shape(test images, best color labels, best color percentages,shapes, 8), on shapes és ["Flip Flops"] i 8 el nombre d'imatges que volem visualitzar. I obtenim:

Figure 4: Flip Flops

2.1.3 Retrieval combined

Aquesta funció és simplement la combinació del Retrieval_by_color i el Retrieval_by_shape, podent així filtrar les imatges per color i forma. Per a provar aquest mètode haurem d'inicialitzar el booleà Retrieval_combined.

És dins de la funció Retrieval_combined que cridarem els dos retrievals anteriors, que ens tornaran una llista de cada una d'elles i seguidament calcularem la seva intersecció, obtenint l'índex de totes les imatges de les formes i colors que busquem.

Utilitzem els paràmetres:
Retrieval_combined(imgs, predicted_class_lables, predicted_class_percentatges, shapes, best_color, best_color_percentatges, colors, 8), on shapes és ["Handbags"], colors és ["Red"] i 8 el nombre d'imatges que volem visualitzar. Obtenim el següent:

Figure 5: Red and Handbags

2.2 Funcions d'anàlisi quantitatiu

Les funcions d'anàlisi quantitatiu són aquelles que ens permeten avaluar de forma numèrica els nostres classificadors. A partir d'aquestes funcions volem veure si les etiquetes que hem assignat referents al color i forma mostren coherentment el que hi ha a la imatge real. Les funcions que ens permeten obtenir aquesta informació són les següents: Kmean_statistics, Get_shape_accuracy i Get_color_accuracy.

2.2.1 K-Means statistics

La funció rep com a paràmetres una classe K-Means amb un conjunt d'imatges i un valor Kmax i ens retorna 4 estadístiques. Els paràmetres en els que es centra són els següents: a l'eix de les absisses es troba el número dels clusters (K) i a l'eix de les ordenades pot aparèixer o bé la Withinclassdistance, les iteracions, el temps d'execució o bé la Intraclassdistance.
Utilitzem aquestes estadístiques per veure com varia cada un d'aquests aspectes a mesura que augmentem la K (número de clusters). Com que serveix per fer un anàlisi de l'algoritme necessitarem inicialitzar un objecte K-Means. Un cop inicialitzat, obtenim el següent gràfic:

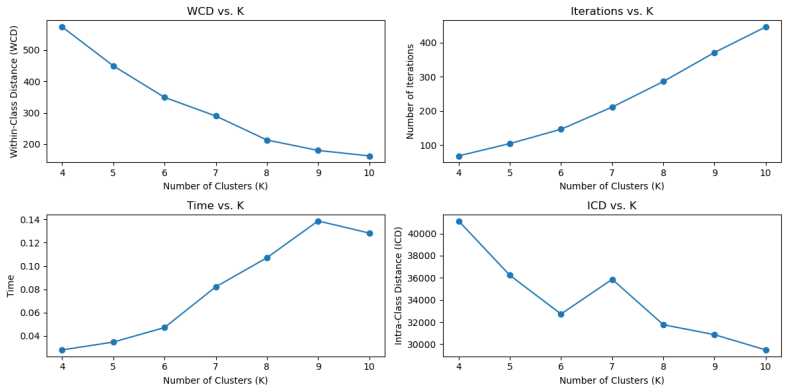

Figure 6: K-Means statistics

En aquest gràfic podem veure que tant el temps com les iteracions tenen una tendència a l'alça cosa que té sentit ja que a més K, ha de fer més iteracions i per tant, implica més temps. A més a més, podem observar que la WCD i la ICD disminueixen. La WCD ho fa des d'un principi, en canvi la ICD ens agradaria que fos el més gran possible, però

podem observar que acaba decreixent. Això es deu al fet que arriba un punt que si fem tants clusters no ens hi caben a l'espai, i per tant es reduirà la distància entre ells.

2.2.2 Get color accuracy

Aquesta funció ens retornarà el percentatge d'encerts del color. Cada imatge pot tenir més d'un color, per tant, inicialitzem un objecte K-Means cridant la funció find_bestK(), ja que ens donarà una predicció més encertada dels colors.

Per calcular el percentatge, calcularem la raó entre els colors ben trobats i els certs (Ground Truth).

Accuray obtinguda: 40.000000000000014 %

Hem comprovat el seu funcionament utilitzant el mateix K-Means que en la funció Retrieval_by_color. Veiem que el percentatge d'encerts no és massa alt. No obstant, considerem que es una bona xifra, ja que estem fent servir un mètode que ens busca una precisió molt alta. És a dir, estem mirant que hagi encertat absolutament tots els colors i ens podem trobar que una imatge al nostre test ens surti amb més colors del Ground Truth, ja que té un número de píxels insignificants d'aquell color.

2.2.3 Get shape accuracy

Aquesta funció, igual que l'anterior, ens indicarà numèricament com d'encertada és la predicció que nosaltres fem de la forma de la roba respecte la real.

Rep una llista d'imatges amb la forma escollida que hem trobat i el seu Ground Truth, i el que fa és mirar cada imatge de l'etiqueta trobada i comparar-la amb l'etiqueta real. Seguidament, per trobar el percentatge (accuracy) cal calcular quantes etiquetes que hem obtingut nosaltres són correctes respecte les que ens han passat com a correctes (Ground Truth). Finalment, ens retornarà un percentatge, el qual és l'accuracy que hem encertat. Com més alt sigui el percentatge, més etiquetes coicidiran.

Accuray obtinguda: 91.30434782608695 %

Hem comprovat el seu funcionament utilitzant el mateix KNN que en la funció Retrieval_by_shape. Veiem que ens dona una accuracy molt alta i per tant, podem considerar que hem fet un bon KNN.

3 Millores sobre K-Means i KNN

3.1 Inicialització de centroides

Hem fet quatre inicialitzacions per tal de tenir algun criteri per poder inicialitzar els centroides a més de canviar el diccionari options, més concretament a l'apartat options['km init'], per poder així inicialitzar el K-Means correctament.
La primera inicialització que hem creat ha estat l'opció 'first'. La qual assigna als centroides els primers K punts de la imatge X que siguin diferents entre ells, i seguidament hem establert la condició options['km_init'] = 'first'. La segona ha estat l'opció random (options['km_init'] = 'random'). Aquestes dos van estar implementades en la primera part del projecte.

A més a més, hem volgut afegir dues noves maneres d'inicialitzar-los, per veure si era millor que les que ja teniem, aquestes noves inicialitzacions han estat les opcions last (options['km_init'] = 'last') i extreme (options['km_init'] = 'extreme').

L'opció 'last', segueix un criteri de selecció semblant a l'opció 'first', però a l'inversa, ja que assigna als centroides els darrers K punts de la imatge X que siguin diferents entre ells.

L'opció 'extreme' distribueix els centroides de manera extrema. És a dir, agafa els punts que major distància tenen entre ells i els va afegint a la llista.

Seguidament hem comparat els seus resultats mirant quin percentetge d'etiquetes predites eren certes. Els paràmetres amb que inicialitzem el K-Means són:

- Cada imatge de test images
- K = 4
- options = 'km_init': option, 'fitting': 'WCD' on option és 'first', 'random', 'last' i 'extreme'

Resultats obtinguts:
Accuracy on color prediction, using option first: 40.000000000000014%
Accuracy on color prediction, using option random: %
Accuracy on color prediction, using option last: 39.44444444444446%
Accuracy on color prediction, using option extreme: 27.361111111111114%

L'opció random no acaba de funcionar bé, perquè intenta calcular el mean d'una llista buida.

A partir d'aquests resultats, podem dir que la millor opció seria utilitzar l'opció first, ja que ens proporciona una millor accuracy.

3.2 Addició d'heurítiques per Find BestK

Per tal de veure si millora la precisió de l'algorisme hem decidit afegir diferents heurístiques per a trobar la millor K. A més de la WCD (withinClassDistance) ja programada, hem implementat dues funcions noves dins de la classe K-Means anomenades ICD (interClassDistance) i FC (FisherCoefficient), les quals calculen la distància inter-class de l'actual cluster i la raó entre WCD i ICD respectivament.

A l'afegir aquestes noves funcions ha estat necessari modificar la funció find_bestK, ja que haviem de fer possible l'us de les noves heurístiques. Per fer-ho, hem utilitzat l'opció 'fitting' del diccionari options i hem establert noves condicions: options['fitting'] = WCD, options['fitting'] = ICD i options['fitting'] = FC.

A continuació hem fet un análisi de les tres heurístiques, resumit en el següent gràfic:

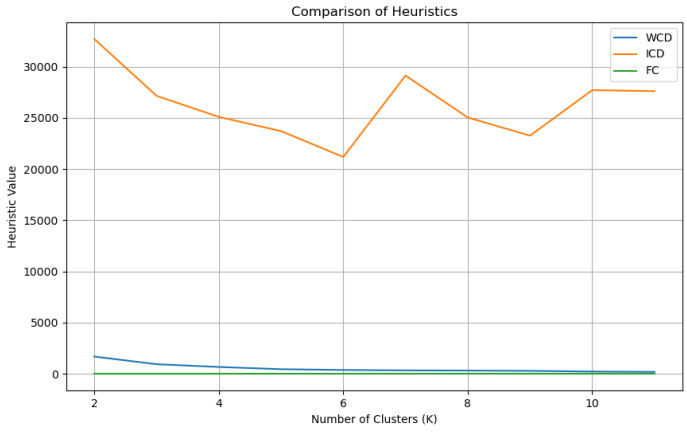

Figure 7: Comparison of Heuristics

En aquest gràfic es mostra com varia la K (des de 2 fins a 11), utilitzant cadascuna de les tres heurístiques . Els diferents valors han estat: Best_K using WCD: 11, Best_K

using ICD: 2, Best_K using FC: 11. Tots els valors obtinguts tenen sentit, tenint en compte que FC i WCD han de minimitzar-se i ICD ha de maximitzar-se, segons la teoria que hem vist a classe.

3.3 Funció Find BestK

Aquesta funció té com a objectiu trobar la millor K possible utilitzant diferents valors de llindar. Hem fet un gràfic comparant cada valor de llindar (entre 10% i 90%) amb el valor de la K obtingut. Per poder realitzar aquestes proves, hem modificat la funció find_bestK() perquè també hagi de rebre com a paràmetre el valor de llindar, a part de la max_K.

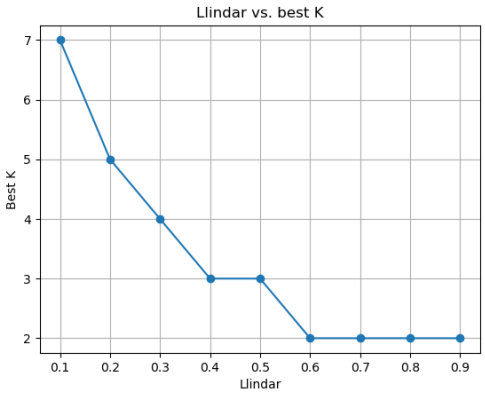

Figure 8: Llindar vs. best K

Després d'observar els resultats, hem arribat a la conclusió que la millor k hauria de tenir un valor ni molt gran, ni molt petit. Si la k resultés ser massa gran, podriem dificultar la interpretació dels resultats. En canvi, si fos massa petita, és possible que l'anàlisi dels resultats no sigués massa exacte. Per això, el millor seria agafar una k mitja. En aquest cas, el millor sería agafar k=5 o k=4, el que equivaldria a un llindar del 20% o 30% respectivament.

3.4 Features for KNN

Per millor l'algorisme KNN, hem creat una nova funció anomenada image_to_grey(), la qual treu el color dels objectes KNN passant les imatges a escala de grisos usant la funció rgb2gray proporcionada en el fitxer utils.py. Cal comparar els resultats obtinguts amb els resultats obtinguts sense aplicar la funció mencionada. Fem dos tests, un per cadascuna de les dues situacions mencionades. En el primer inicialitzem l'objecte KNN amb un array que conté les imatges passades a escala de grisos i una llista amb les etiquetes de les imatges. En el segon, l'inicialitzem amb un array que conté les imatges de prova i la mateixa llista d'etiquetes.

Resultats obtinguts:

Retrieval_by_shape time: 5.764359712600708
Features_KNN time: 2.5493485927581787

Podem veure clarament com el nostre KNN millora aplicant aquesta funció.

3.5 Unique colors percentages

Hem creat una nova funció dins del fitxer my labeling que permet prendre la llista de colors i l'array de percentatges retornats per la funció get_colors i fer el següent: Primer, convertir la llista de colors en un array sense repeticions (unique colors) utilitzant la comanda de numpy np.unique(). A continuació, per a cada color, hem fet la mitjana dels seus percentatges, aconseguint un array de la mateix llargada que unique colors que segueixen el mateix ordre.

4 Conclusions

Durant aquest treball, hem explorat l'ús dels algoritmes K-Means i KNN per a l'etiquetatge automàtic d'imatges de roba segons el tipus de peça i color. A partir de les nostres investigacions i experimentacions, hem arribat a les següents conclusions:

La inicialització dels centroides dels clusters amb l'opció options['km_init'] = 'first' en l'algoritme K-Means pot ajudar a millorar el rendiment.
Aquesta inicialització utilitza els centroides dels clústers de l'iteració anterior com a punt de partida per a l'iteració actual, fent que la convergència sigui més eficient.
Utilitzar tres heurístiques és significativament millor que utilitzar-ne només una, ja que prediu un millor valor de K.
Passar com a paràmetre el percentatge del color desitjat a les funcions retrieval_by_color i retrieval_by_shape, fa que obtinguem un millor resultat.
Emprar les imatges amb els píxels de color gris per a cada imatge, ens disminueix significativament el temps d'execució.

És important considerar aquests factors al seleccionar l'algoritme en funció de les necessitats del sistema.

No obstant, també ens hem trobat amb diversos problemes al llarg del desenvolupament de la pràctica. Millorar la funció find_bestK ha estat prou difícil, en especial, a l'intentar buscar el millor llindar. Per altra banda, com s'ha comentat anteriorment, després d'haver imprementat la ICD, hem observat que el seu valor decreix a mesura que les K augmenten, per lo que, segons la teoria vista a classe, arribem a la conclusió que la nostra funció no és lo prou bona possible.