TP : Classification d'images avec des descripteurs locaux

Univ. Lille – Master Informatique, parcours RVA, UE AIV2

1 Objectifs

On s'intéresse dans ce TP à la classification d'images. L'objectif est de mettre en place des pipelines de classification d'images basés sur des descripteurs locaux (histogrammes de gradient SIFT) et un classifieur supervisé. Deux méthodes d'agrégation de descripteurs locaux seront mises en œuvre : les sacs de mots visuels et les VLAD.

2 Données

Un sous-ensemble du jeu de données Caltech-101 sera utilisé. Il contient 1000 images réparties en 10 classes (100 images / classe) et les labels (classes) associés à ces images. Chaque image contient un objet principal correspondant à son label.

Le jeu de données est fourni dans l'archive `caltech101_subset.tar.gz` disponible sur Moodle. L'archive contient :

— les images du jeu de données, dans le répertoire `caltech101_subset`,

— la liste des 1000 images à utiliser et leur label associé, dans le fichier (texte) `caltech-subset.files`.

Des vocabulaires visuels précalculés sont fournis dans l'archive `vocabulaires_sift.tar.gz` disponible sur Moodle. Les vocabulaires sont des tableaux Numpy (un centroïde par ligne) enregistrés au format `.npy`. Chaque fichier est nommé `vocabulary_n.npy`, où `n` est la taille (nombre de centroïdes) du vocabulaire. Les vocabulaires visuels ont été obtenus en utilisant l'algorithme de clustering k-means sur 10^6 descripteurs SIFT échantillonnés sur le jeu de données Holidays.

3 Logiciels

L'implémentation du TP se fera en Python. Les bibliothèques suivantes seront utilisées :

— OpenCV (package `cv2`), pour les fonctions de base sur les images (ouverture, conversion, etc.) et le calcul des descripteurs SIFT,

— Scikit-learn (package `sklearn`), pour la partie classification supervisée et la quantification des descripteurs,

— Numpy (package `numpy`), pour la manipulation des vecteurs / matrices,

— Matplotlib (package `matplotlib.pyplot`) pour la visualisation,

— le module `vlad.py`, disponible sur Moodle, pour le calcul des descripteurs VLAD.

4 Descripteurs SIFT

Dans cette partie, nous commencerons par détecter des régions d'intérêt avec le détecteur *difference-of-Gaussians* (*détecteur* SIFT) et décrire ces régions avec des histogrammes d'orientation de gradient (*descripteur* SIFT).

Questions

1. Reprenez votre code pour lire les images du jeu de données et les convertir en niveaux de gris du TP précédent.
2. Écrivez une fonction qui retourne les points d'intérêt et descripteurs SIFT sur l'ensemble des images du jeu de données. La fonction retournera un couple de tableaux Numpy, le premier contenant les listes de points d'intérêt des images, le second les matrices de descripteurs.
3. Écrivez une fonction qui permet d'afficher les points d'intérêt détectés sur une image.
4. Visualisez les points d'intérêt détectés sur des échantillons d'images de chaque classe.

Fonctions / classes utiles

— `SIFT_create()`, fonction du module `cv2.xfeatures`, qui permet d'instancier un objet de la classe `xfeatures2d_SIFT` pour détecter et décrire des points d'intérêt SIFT,

— `detectAndCompute()`, méthode de la classe `xfeatures2d_SIFT` qui détecte et décrit les points d'intérêt SIFT dans une image,

— `Keypoint`, classe du module `cv2` qui donne les propriétés d'un point d'intérêt (position, taille, etc.),

— `circle()`, fonction du module `cv2`, qui permet de dessiner un cercle dans une image,

— `round()`, fonction du module `numpy`, qui permet d'arrondir des valeurs réelles.

5 Sacs de mots visuels

Dans cette partie, nous allons implémenter les sacs de mots visuels. Pour rappel, la construction du sac de mots visuels d'une image suit les étapes suivantes :

1. Construction d'un vocabulaire visuel. Ici, le vocabulaire est pré-calculé et fourni dans les données du TP.
2. Détection et description des points d'intérêt dans l'image à décrire.
3. Assignation des descripteurs à leur mot visuel le plus proche dans le vocabulaire.
4. Construction de l'histogramme d'occurrences des mots visuels.
5. Normalisation de l'histogramme.

On utilisera des descripteurs SIFT et un vocabulaire de 5000 mots visuels.

Questions

1. Écrivez une fonction qui retourne l'histogramme de mots visuels d'une image.
2. Écrivez une fonction qui calcule les histogrammes de mots visuels pour l'ensemble des images du jeu de données.

Fonctions / classes utiles

— `load()`, fonction du module `numpy`, qui permet de lire les fichiers `.npy`,

— `zeros()`, fonction du module `numpy`, qui permet d'initialiser un nouveau tableau contenant des 0,

— `KMeans`, classe du module `sklearn.clustering`, qui permet d'effectuer le clustering d'un ensemble de vecteurs et d'assigner des vecteurs à leur centroïde le plus proche. L'attribut `cluster_centers_` de la classe permet d'obtenir ou d'initialiser les centroïdes du quantifieur. Alternativement, on peut utiliser la classe NearestNeighbors du module `sklearn.neighbors` pour réaliser l'assignation des vecteurs à leur centroïde le plus proche,

— `unique()`, fonction du module `numpy`, qui permet d'obtenir les éléments uniques (et plus...) d'un tableau Numpy,

— le slicing par indice sur les tableaux Numpy,

— `sum()`, fonction du module `numpy`, qui permet de calculer la somme des éléments d'un tableau Numpy.

6 Classification par sacs de mots visuels

On va maintenant mettre en place le système de reconnaissance d'images basé sur les sacs de mots visuels. On utilisera le même protocole de classification que pour le TP préceédent :

— classifieur de type régression logistique,

— 80% des données pour l'apprentissage et 20% de données pour le test,

— évaluation du taux de reconnaissance (*accuracy*),

— graine aléatoire fixe pour obtenir des résultats comparables d'une expérimentation à une autre.

Questions

1. Écrire une fonction qui effectue le processus complet d'apprentissage et de test et retourne le taux de reconnaissance sur l'ensemble de test.
2. Quel est le taux de reconnaissance obtenu ?
3. Comparez le taux de reconnaissance obtenu avec les valeurs obtenues précédemment avec les histogrammes de couleur et les histogrammes de LBP et interprétez ces résultats.
4. Calculez le taux de reconnaissance obtenu avec les autres vocabulaires fournis. Interprétez les performances observées.

Fonctions / classes / méthodes utiles

— `LogisticRegression`, classe du module `sklearn.linear_model`, qui implémente le classifieur régression logistique,

— `train_test_split()`, fonction du module `sklearn.model_selection`, qui sépare un jeu de données en ensembles d'entraînement et de test,

— `fit()`, méthode des classifieurs Scikit-learn, qui effectue l'entraînement du classifieur,

— `predict()`, méthode des classifieurs Scikit-learn, qui retourne les prédictions du classifieur entraîné pour un ensemble de test,

— `accuracy_score()`, fonction du module `sklearn.metrics`, qui calcule le taux de reconnaissance.

7 Implémentation des VLAD

On s'intéresse maintenant à une seconde méthode d'agrégation des descripteurs locaux : le VLAD. Pour rappel, le VLAD d'une image s'obtient via les étapes suivantes :

1. Construction d'un vocabulaire visuel. Ici, le vocabulaire est pré-calculé et fourni dans les données du TP.
2. Détection et description des points d'intérêt dans l'image à décrire.
3. Assignation des descripteurs à leur mot visuel le plus proche dans le vocabulaire.
4. Pour chaque mot du vocabulaire visuel, calcul de la somme des résidus des vecteurs, produisant un vecteur de la même dimension que les descripteurs locaux / centroïdes.
5. Concaténation des vecteurs précédents en un unique vecteur.

6. Normalisation en racine carrée des composantes du vecteur VLAD.
7. Normalisation en norme euclidienne du vecteur VLAD.

Il est également possible de réduire la dimension des VLAD calculés sur les images de la collection grâce à une ACP.

Le code permettant de calculer le descripteur VLAD d'une image à partir des descripteurs SIFT extraits de celle-ci est fourni dans le module `vlad.py`.

On utilisera dans ce TP des descripteurs SIFT et un vocabulaire visuel de taille 50. Pour l'ACP, on conservera les 100 composantes principales.

Questions

1. Quelle sera la dimension du VLAD d'une image avec des descripteurs SIFT et un vocabulaire de taille 50 ?
2. Écrivez une fonction qui calcule les vecteurs VLAD pour l'ensemble des images du jeu de données.
3. Écrivez une fonction qui permet de réduire la dimension des vecteurs VLAD par ACP. Il sera nécessaire de centrer les vecteurs avant d'appliquer l'ACP.

Fonctions / classes utiles

— `load()`, fonction du module `numpy`, qui permet de lire les fichiers `.npy`,

— `zeros()`, fonction du module `numpy`, qui permet d'initialiser un nouveau tableau contenant des 0,

— `vlad()`, fonction du module `vlad` fourni sur Moodle, qui permet de calculer le VLAD d'une image à partir de ses descripteurs SIFT,

— les opérateurs arithmétiques de `numpy`,

— `mean(()`. fonction du module `numpy`, qui permet de calculer des moyennes dans un array Numpy,

— `PCA`, classe du module `sklearn.decomposition`, qui permet de réaliser une ACP.

8 Classification par VLAD

On va maintenant reproduire les expérimentations précédentes en utilisant les VLAD comme descripteurs.

Questions

1. Mettez en place un pipeline de reconnaissance d'images similaire au précédent, mais reposant maintenant sur les VLAD (sans réduction de dimension).
2. Calculez le taux de reconnaissance (*accuracy*).
3. Comparez les performances du VLAD à celles des sacs de mots visuels. Comment expliquer cette différence de performance ?
4. Effectuez les mêmes expérimentations en utilisant maintenant l'ACP pour réduire la dimension des vecteurs VLAD à 100 dimensions.
5. Quel est le principal avantage d'utiliser une réduction de dimensions ici ?

9 Découpage géométrique des images

Comme pour les descripteurs globaux, il est possible d'effectuer un découpage géométrique de l'image. On calcule alors des sacs de mots visuels ou des VLAD sur chacune des régions, puis on les concatène pour obtenir un unique vecteur descripteur.

Questions

1. Adaptez votre code précédent pour intégrer le calcul de sacs de mots visuels et de VLAD sur les régions d'une grille 5×5.
2. Reproduisez le processus d'apprentissage avec ces nouveaux descripteurs et mesurez les taux de reconnaissance (*accuracy*) obtenus sur l'ensemble de test.