ENSTA-Paris 2e année - Cours MI204
TP2 : Classification de caractéristiques locales
Approches supervisée (bayésienne) et non supervisée (K-Means)

L'objectif de ce TP est de comprendre le principe de l'apprentissage bayésien et celui du groupement non supervisé, et de l'appliquer à un problème de segmentation d'images par classification de pixels pour une application de votre choix.

1 Préliminaires informatiques

1.1 Démonstration : Inti

Le logiciel utilisé dans la première partie du TP (Figure 1) permet de visualiser à la fois les images et le modèle bayésien correspondant à une partition d'un espace d'observation de dimension 2 en 2 classes.

FIGURE 1 – L'interface *Bayes_Classif*.

L'utilisation de ce logiciel n'est pas indispensable à la réalisation du TP. Ceux qui souhaitent l'utiliser peuvent néanmoins le télécharger à partir du lien suivant :

`http://perso.ensta-paris.fr/~manzaner/Cours/MI204/Bayes_Classif.zip`

1.2 Expérimentation et développement : Python

Dans la deuxième partie du TP, on vous demande d'expérimenter et développer vos propres solutions en utilisant Python avec Scikit-Learn et OpenCV.

Une base de code est fournie, qui fonctionne avec une installation standard d'OpenCV (sans opencv-contrib, version testée 4.1.0) et Python3. L'archive contenant cette base peut être téléchargée au lien suivant :
`https://perso.ensta-paris.fr/~manzaner/Cours/MI204/TP2_Python_Classif.zip`

Les packages / modules suivants sont nécessaires :
— python-opencv
— numpy
— matplotlib
— sys
— scikit-learn
Tous les codes fournis sont des scripts Python3 éditables et exécutables en lançant la commande, par exemple :

```
python3 Display_Components.py <Mon_image.jpg>
```

On trouvera quelques images d'expérimentation dans l'archive suivante :
`https://perso.ensta-paris.fr/~manzaner/Cours/MI204/Images_Classif.zip`
Les séries d'images fournies, correspondent à trois problèmes différents :
— *INRA :* Images de feuilles de blé attaquées par un champignon. Il s'agit d'évaluer l'efficacité d'un fongicide en calculant automatiquement la proportion de la feuille occupée par le champignon.
— *Essex :* Visages. Il s'agit d'isoler les pixels de la peau du reste de l'image, de façon à localiser précisément le visage sur l'image.
— *Kitti :* Scènes routières en milieu forestier. Il s'agit d'isoler les pixels de la route de façon à segmenter la voie navigable dans un objectif de conduite assistée / autonome.

OpenCV et Scikit-Learn sont des bibliothèques très utilisées en analyse d'images et en apprentissage automatique. On pourra consulter avec profit de nombreux guides et tutoriels en ligne (attention toutefois aux versions fournies dans les exemples). On se limitera ici à mentionner les sites officiels pour le guide de référence des fonctions :
`https://docs.opencv.org/4.1.0/`
`https://scikit-learn.org/stable/user_guide.html`
ainsi que les tutoriels :
`https://opencv24-python-tutorials.readthedocs.io/en/stable/index.html`
`https://scikit-learn.org/stable/tutorial/index.html`

2 Principes de la classification bayésienne

2.1 Échantillonnage manuel

On va d'abord expérimenter le principe de la classification bayésienne sur des nuages de points 2d modélisées selon des distributions gaussiennes. On lance pour cela le script :

```
./Bayes_Cloud.tcl
```

Ce script permet d'entrer à la main les points de l'espace d'observation 2d, en cliquant dans la fenêtre principale. Un clic gauche permet d'entrer un point de la classe 1 (représentée en bleu), et un clic droit un point de la classe 2 (représentée en rouge). Un curseur permet d'ajuster la valeur de la probabilité *a priori* sur la classe 1.

QUESTIONS : Quelle différence y a-t-il entre une classification "euclidienne" naïve et la classification bayésienne ? Comment peut-on voir la différence entre le critère du maximum de vraisemblance (ML) et celui du maximum *a posteriori* (MAP) ?

2.2 Application à la classification bayésienne des couleurs

L'objectif est d'appliquer la classification à des pixels d'une image couleur, où l'espace d'observation est un vecteur à 2 composantes, lesquelles sont issues d'un des espaces couleurs classiques : (R,G,B), (H,S,V) ou (Y,Cr,Cb). Dans un premier temps on va observer et interpréter chacune de ces composantes en lançant le script :

```
>./Bayes_Classif.tcl <mon_image_couleur.png>
```

On affichera les composantes à partir du menu *Colour* → *Display...*. Il est conseillé de travailler sur des couleurs vives, on pourra par exemple utiliser l'image suivante :

`https://perso.ensta-paris.fr/~manzaner/Cours/MI204/Parrots.jpg`

QUESTIONS : Que représentent ces différentes composantes ? Quelles sont leurs propriétés en termes d'invariance ? Quelles sont leurs éventuelles limites en termes numériques ? En projetant quelques pixels de l'image dans l'espace d'observation (Sélectionner l'espace désiré dans le menu *Classif* → *Set Space...*), peut-on observer des corrélations entre composantes ?

NB : Cette partie nécessite quelque documentation sur la couleur. On pourra (par exemple !) se référer au complément du 2e cours :

`http://perso.ensta-paris.fr/~manzaner/Cours/MI204/2bis_Couleur.pdf`

3 Classification de caractéristiques locales

Dans la suite du TP on vous demande de choisir un problème de segmentation d'images par classification de caractéristiques locales, soit dans l'un des trois problèmes présents dans la base d'images, soit en proposant votre propre problème, et en développant votre solution à partir des codes Python fournis dans la base logicielle.

3.1 Classification bayésienne des pixels

Vous pouvez dans un premier temps créer et observer vos caractéristiques locales à partir du script `Display_Components.py` (par défaut, le script affiche les composantes couleur : RGB, puis HSV, puis YCbCr).

Puis vous pouvez utiliser ces caractéristiques pour créer votre modèle bayésien en utilisant le script `Bayes_Model_Training.py`, qui vous permet de superviser le modèle en entrant des exemples (par défaut on est sur 2 classes : positif "p" et négatif "n") sous la forme de "batches" de pixels à l'aide de la souris (dessiner plusieurs RoI de pixels exemples) et du clavier (touche "p" et "n"). Lorsque votre apprentissage est terminé, vous appuyez sur la touche "q" pour quitter.
Vous pouvez alors tester votre modèle sur l'image de votre choix (par défaut c'est fait sur l'image d'apprentissage).

QUESTIONS : Quelles vous semblent être les espaces d'observations et le nombre de classes les mieux adaptés à la tâche qui vous intéresse ? Quelles sont les difficultés potentielles ? Quelles différences observez-vous entre le modèle bayésien gaussien multidimensionnel et le modèle bayésien gaussien naïf ? Quelles caractéristiques locales (au-delà des composantes couleur) pourraient être utilisées avec profit pour améliorer votre modèle ?

3.2 Classification non supervisée des pixels par K-Means

Comparez l'approche et les résultats précédents à ce qu'on peut obtenir avec une méthode non supervisée de clustering : l'algorithme K-Means. Le script `KMeans_Clustering.py` vous permet de créer le clustering sur l'image de votre choix, d'afficher le centre des clusters (par défaut : 6 classes), puis d'afficher les labels correspondants sur l'image de votre choix.

QUESTIONS : Quelles vous semblent être les espaces d'observations et le nombre de classes les mieux adaptés à la tâche qui vous intéresse ? Quelles sont les difficultés potentielles ? Quels sont les avantages et les inconvénients par rapport à l'approche précédente ?

4 Remarques générales

Comme pour les autres TP, vos arguments, critiques et discussions ont plus d'importance dans l'évaluation que vos résultats. Pensez à illustrer votre rapport avec des résultats de traitement.