

SORBONNE UNIVERSITÉ SCIENCES

RAPPORT RDFIA DES TMEs 1, 2 ET 3

SIFT, Visual Dictionary, Bag of Words, Classification

MASTER 2 DONNÉES, APPRENTISSAGE ET CONNAISSANCES



OUARDA FENEK

2019 - 2020

1 Réponses aux questions du TME 1_2

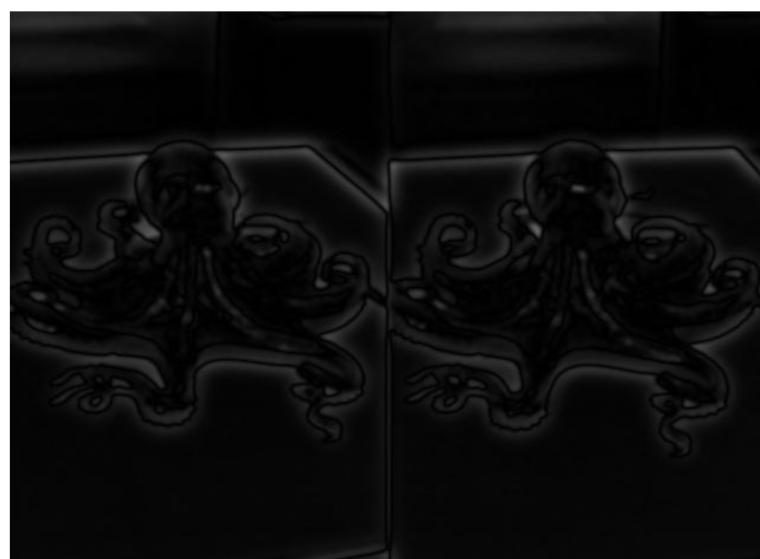
1. On a : $h_x = [x1 \ x2 \ x3]^T$, $h_y = [y1 \ y2 \ y3]^T$ et on a

$$\begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * \begin{bmatrix} -1 & 0 & +1 \end{bmatrix} \quad \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ +1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

Par identification on aura : $h_y = [0.25 \ 0.5 \ 0.25]^T$ et $h_x = [-0.25 \ 0 \ 0.25]^T$.

2. On sépare les filtres, pour avoir moins de calculs à faire lors des implémentations.
Avec la division par quatre, on montre la dérivation dans une direction donnée de l'image est liée à un lissage triangulaire dans l'autre direction qui sert à éliminer les faux contours.
3. La pondération par le masque gaussien permet d'éliminer certains détails jugés insignifiants, laisser uniquement les caractéristiques importantes de l'image pour ne pas perdre en généralité. Cependant il faut aussi faire attention à ne pas tout enlever jusqu'à ce que ça devient insignifiant.

Exemple :



- La descrétisation des orientations consiste à attribuer à chaque point-clé une ou plusieurs orientations déterminées localement sur l'image à partir de la direction des gradients dans un voisinage autour de ce point.

Et comme les descripteurs sont calculés relativement à ces orientations, la descrétisation est essentielle pour assurer l'invariance de ceux-ci à la rotation. En effet les mêmes descripteurs doivent être obtenus à partir de la même image quelque soit son orientation.

- Le post-processing qui se fait en plafonnant les valeurs à 0.2 et en normalisant le descripteur est dans le but de diminuer la sensibilité du descripteur aux changements de luminosité. En effet, on veut que si on change de luminosité on arrive toujours à obtenir à peu près les mêmes points.

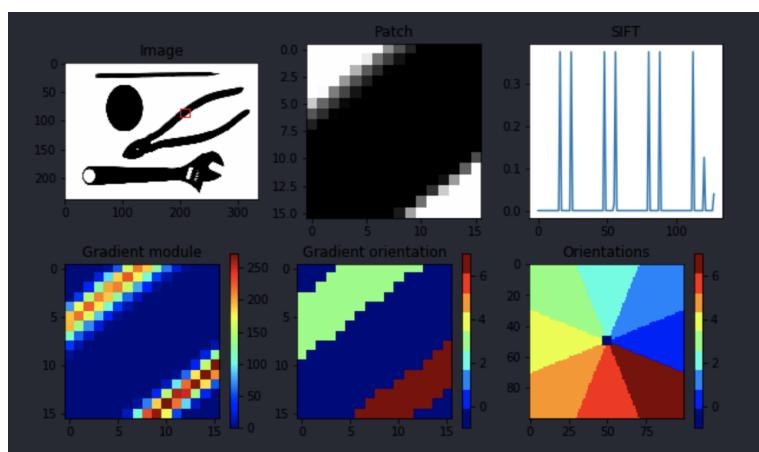
Exemple :



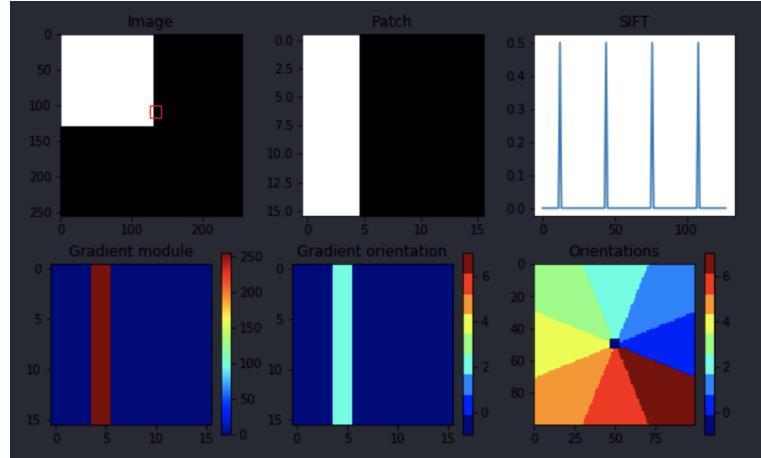
Malgré la rotation et l'exposition, le sift devrait renvoyer le même descripteur pour ces deux images.

- L'idée et l'intérêt de SIFT : L'algorithme consiste à calculer que l'on appelle des descripteurs SIFT. Ce sont des informations numériques dérivées de l'analyse locale d'une image. Ils caractérisent son contenu visuel le plus indépendamment possible de la résolution du capteur de cette image, du zoom, de la luminosité et de l'angle d'observation. C'est ici que réside son succès, deux photos différentes d'un même objet auront des descripteurs quasi-similaires même si les circonstances de prise de ces photos sont différentes. D'un autre côté, deux photos d'objets très différents, produiront deux descripteurs très différents aussi. Cet algorithme a un pouvoir descriminant.

- Intérprétation des résultats :



Pour cette zone de l'image, on voit bien qu'il y a pas mal de valeurs dans le descripteur qui ne sont pas nulles ça s'explique par le fait que les bordures s'étale sur les deux axes du patch. En visualisant la fenêtre d'orientations des gradients on voit qui se passe quelque chose aux bordures (l'orientation du gradient n'est pas nulle) et dès qu'on est dans une zone homogène elle l'est. Pareil pour le module du gradient, il prend valeur qu'aux bordures (là où il se passe quelque chose, là où il y a un changement).



Par rapport au patch précédent il y a plus de valeurs nulles dans le sift. ça s'explique par le fait qu'il y a moins de changement dans l'image, il y a une seule bordure à trouver sur un seul axe.

8. Un dictionnaire est nécessaire dans le processus que nous essayons de mettre en place car nous avons un très grand nombre d'images, voire d'objets, et nous voulons réduire celà. La création de ce dictionnaire permettra par exemple de mettre deux sifts qui représentent le même objet, ou deux objets qui se ressemblent, dans une même classe.

Exemple :



9. Etant donné un nuage de points x_i pondérés par des masses m_i , le barycentre x_c est le point qui minimise la somme des carrés pondérés des distances aux x_i , ie, $\min_c \sum_i^n m_i \|x_i - x_c\|^2$. Par définition aussi, pour retrouver les coordonnées de ce barycentre : $x_c = \frac{\sum_i^n x_i}{\sum_i^n m_i}$ et comme dans

notre cas, tout les points sont pondérés par 1, i.e. $m_i = 1$ quelque soit i , on a alors : $x_c = \frac{\sum_i^n x_i}{n}$, d'où le centre du cluster qui minimise la dispersion est bien la moyenne des points x_i .

Autrement :

Quelque soit a :

$$\begin{aligned} \sum_i^n (x_i - a)^2 &= \sum_i^n (x_i - \bar{X} + \bar{X} - a)^2 \\ &= \sum_i^n ((x_i - \bar{X}) + (\bar{X} - a))^2 \\ &= \sum_i^n [(x_i - \bar{X})^2 + 2(x_i - \bar{X})(\bar{X} - a) + (\bar{X} - a)^2] \\ &= \sum_i^n (x_i - \bar{X})^2 + 2(\bar{X} - a)(\sum_i^n x_i - n\bar{X}) + n(\bar{X} - a)^2 \\ &= \sum_i^n (x_i - \bar{X})^2 + 2(\bar{X} - a)(\sum_i^n x_i - n\frac{1}{n} \sum_i^n x_i) + n(\bar{X} - a)^2 \\ &= \sum_i^n (x_i - \bar{X})^2 + n(\bar{X} - a)^2 \end{aligned}$$

Si $\bar{X} = a$ cette distance est minimale car elle serait égale à : $\sum_i^n (x_i - \bar{X})^2$

Si $\bar{X} \neq a$ on aura $\sum_i^n (x_i - \bar{X})^2 < \sum_i^n (x_i - \bar{X})^2 + n(\bar{X} - a)^2$.

10. Choisir un nombre de cluster K n'est pas forcément intuitif. Spécialement quand le jeu de données est grand et qu'on n'ait pas un a priori ou des hypothèses sur les données. Un nombre K grand peut conduire à un partitionnement trop fragmenté des données. Ce qui empêchera de découvrir des patterns intéressants dans les données. Par contre, un nombre de clusters trop petit, conduira à avoir, potentiellement, des cluster trop généralistes contenant beaucoup de données. Dans ce cas, on n'aura pas de patterns "fins" à découvrir.

La méthode la plus usuelle pour choisir le nombre de clusters est de lancer K-Means avec différentes valeurs de K et de calculer la variance des différents clusters. La variance est la somme des distances entre chaque centroïde d'un cluster et les différentes observations incluses dans le même cluster. Ainsi, on cherche à trouver un nombre de clusters K de telle sorte que les clusters retenus minimisent la distance entre leurs centres (centroïds) et les observations dans le même cluster. On parle de minimisation de la distance intra-classe.

11. La visualisation se fait à travers d'exemples de patchs et pas directement car si on prend des sifts qui n'ont rien à avoir avec l'image, le K-means va assigner chaque sift à un centre mais l'image résultante sera complètement déformée.
12. Commentaires sur les résultats : En affichant pour chaque cluster le patch le plus proche chacun à côté de l'autre, on voit une différence entre les patchs, est c'est ça le but, pour pouvoir différencier entre les clusters.
Par contre quand on affiche pour chaque cluster les 50 patchs qui sont le plus proches de lui, les patchs sont presque indistinguables, très ressemblables.
13. Le vecteur z est une représentation numérique de l'image qui permet de déduire le nombre d'objets différents qu'elle contient (le nombre de clusters) et le nombre d'apparition de chaque cluster (la fréquence).
En d'autres termes, pour chaque patch de l'image, après avoir trouvé son sift associé, on trouve dans le dictionnaire visuel le cluster qui lui est le plus proche, on affecte donc une seule classe à chaque patch (c'est ainsi que h est obtenue). Après avoir fait cela à tout les patchs de l'image, on somme verticalement, et on aura ainsi tous les clusters qui apparaissent dans l'image avec leurs nombres d'apparitions, c'est notre vecteur z
14. Résultat :
On a en premier lieu, le cluster associé au sift (On a mis un K=50 pour notre K-means).



on voit qu'il est associé au dernier cluster (celui qui est nul parce qu'on a beaucoup de patchs non significatifs), donc malheureusement on a pas pu voir grand chose.

Avec 1001 clusters :

On voit qu'il y'a 5 clusters concernés. Dans le cadrillage on a 5 couleurs également ou chacune est associée à un cluster.



15. Le codage au plus proche voisin élimine toute hésitation pour un patch donné. Il l'affecte certainement à une seule classe pour éviter toute confusion. ça a un intérêt surtout pour la classification après, le patch est la plus petite unité de l'image, certainement la plus précise donc il vaut mieux enlever la confusion dès le début.

Un autre codage possible, c'est d'affecter un 1 aux k clusters les plus proches. Si par exemple on a deux classes *chat* et *tigre*, le patch en question est à une distance de 0.4 de *chat* et de 0.45 de *tigre*, au lieu de l'affecter directement à *chat*, on pourrait l'affecter aux deux, vu qu'ils sont proches.

16. Le pooling sum permet de compter le nombre d'objets de chaque classe qui se trouve dans l'image. (La fréquence d'apparition de chaque objet de l'image)

On pourrait utiliser un autre pooling qui reflèterait juste la présence ou l'absence d'un objet dans l'image. C'est à dire, si la somme est différente de 0 on affecte 1, sinon, 0.

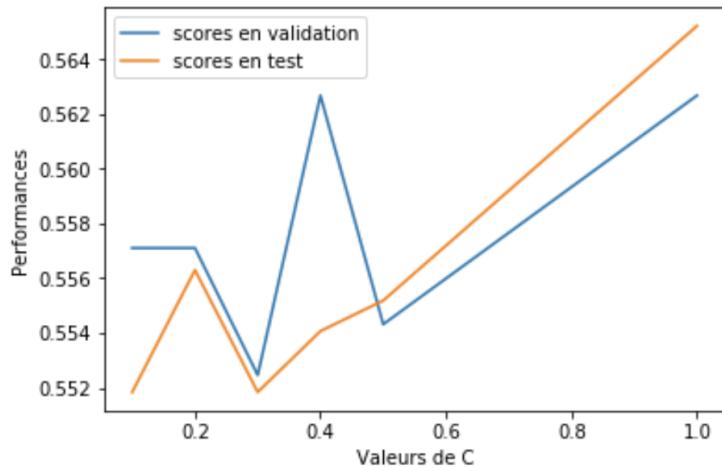
17. On aura que les valeurs de nos vecteurs z seront très variables en magnitudes (en effet, un objet peut apparaître une dizaine de fois et un autre aucune fois dans une même image). Et on sait que plus tard, ces vecteurs seront allimentés à des classifieurs. Mais comme la plupart des algorithmes d'apprentissage automatique utilisent la distance euclidienne entre deux points de données dans leurs calculs, cela pose un problème. Les paramètres d'apprentissage risquent de ne pas converger, c'est pour celà qu'on normalise nos vecteurs.

Une autre normalisation qu'on pourrait utiliser serait la norme L_1 .

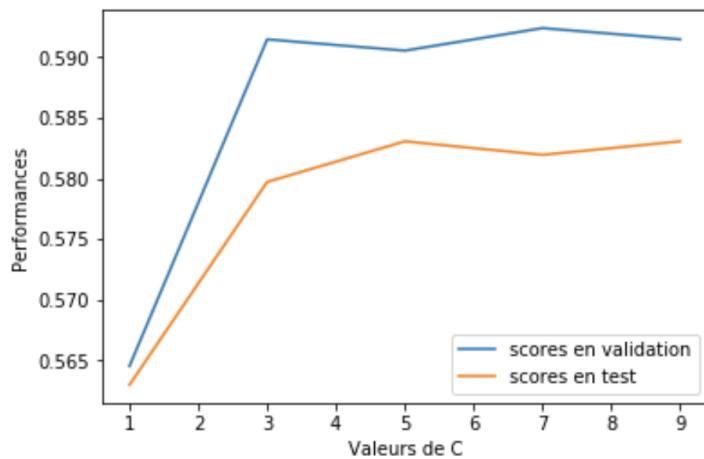
2 Réponses aux questions du TME 3

1. Les courbes de performances :

Sur une échelle [0.1, 0.2, 0.3, 0.4, 0.5, 1] :



Sur une échelle [1, 3, 5, 7, 9] :



Les courbes de validation montrent qu'il est préférable de fixer l'hyper paramètres C à une valeur égale à 7. Pour celle-ci on a eu le score de validation le plus élevé qui est 0.5923862581244197. Le score en test correspondant est 0.5819397993311036.

2. L'ensemble de validation est d'utilité pour fixer les hyper-paramètres de nos modèles de classification.

Quant à l'ensemble de test, il sert à évaluer le modèle, en termes de performances (accuracy, precision, recall ...)

3. Les images en noir et blanc auront le même traitement que les autres, on construira d'abord les sifts qui lui sont associés avec l'algorithme décrit précédemment, (on peut scipper pas mal de prétraitements vu que l'image est déjà en noir et blanc, il y aura besoin d'appliquer beaucoup de masques ou de normalisation).

Une fois les sifts construits, on construit le vecteur Z associé à l'image (l'étape *BoW*) à partir du dictionnaire visuel calculé au préalable. Enfin, on alimente ce dernier à notre programme de classification. Dans ce cas, à notre SVM.

4. Amélioration du processus :

Les méthodes d'extraction de caractéristiques comportent deux phases. La première est la détection des points d'intérêt dans une image. Il n'existe pas de définition formelle de ce qui constitue un point d'intérêt, la plupart des articles le définissant comme une partie intéressante d'une image, des parties facilement reconnaissables dans deux ou plusieurs images différentes. Cette propriété s'appelle la **répétabilité** et sert à mesurer l'efficacité des algorithmes de détection de caractéristiques. Donc, si on veut améliorer le processus d'extraction de features, il est intéressant de considérer cette propriété.

Pour l'améliorer, on pourrait combiner le **SIFT** avec un autre algorithme nommé **FAST**.

Dans la mise en œuvre du **SIFT** classique, les points candidats sont extraits par recherche de points extrema aux alentours de 27 pixels. Cela signifie que chaque pixel de l'image sera comparé à ses 26 voisins (9 à partir de l'image avec un niveau de flou plus faible, 8 à partir de l'image actuelle et 9 à partir de l'image avec un plus grand niveau de flou). Un pixel est considéré comme un point candidat si sa valeur est plus petite ou plus grande que l'ensemble de ses 26 voisins.

FAST est un algorithme de détection de coins dans lequel un cercle de 16 pixels autour d'un pixel candidat p est pris en considération. Le pixel est un coin s'il y a une séquence consécutive de n pixels dans le cercle qui sont tous soit plus lumineux que le pixel candidat par un certain seuil ou plus sombre que le point candidat par le même seuil. Dans notre approche permettant d'augmenter la répétabilité de **SIFT**, au lieu de chercher dans un carré de 3x3 voisins en 3 dimensions, la recherche d'extrema se fait par rechercher dans un cercle de rayon 3 de 16 pixels une séquence de n pixels qui sont tous plus lumineux que le point candidat d'un certain seuil, ou sont plus sombres que le point candidat du même seuil.

Pour améliorer la classification, ils s'avèrent que les méthodes de **SIFT dense** décrites dans [Dalal et Triggs 2015] où il s'agit de calculer des descripteurs SIFT sur chaque point d'une grille préétablie (et donc sans passer par la phase de détection SIFT) sont très efficaces en matière de classification d'images.