Projet 1: Création d'une carte probabilistique d'infarctus cérébral à partir de signaux spatio-temporels d'IRM de perfusion : Approche supervisée non paramétrique

28 Novembre 2017

Référent sujet:

- David Rousseau david.rousseau@univ-angers.fr

1 Mise en contexte

L'accident vasculaire cérébral (AVC) ischémique, ou infarctus cérébral, est causé par l'obstruction d'une artère au niveau du cerveau. L'imagerie par résonance magnétique pondérée de perfusion est une des techniques les plus utilisées dans la prise en charge des patients atteints d'AVC ischémique en phase aiguë pour renseigner sur le débit de la micro-circulation sanguine dans les tissus. Elle consiste à injecter un agent de contraste qui va circuler dans le réseau vasculaire et qui traduit la perfusion tissulaire. La zone atteinte par l'AVC (la lésion ischémique) est hypoperfusée car obstruée par un caillot sanguin (comparée à la perfusion d'un tissu sain).

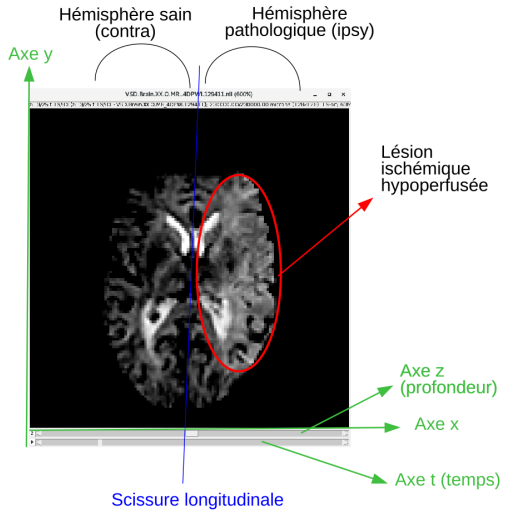

Figure 1: IRM de perfusion d'un patient atteint d'AVC ischémique (à ce temps donné, l'agent de contraste est visible en hypointensité)

L'IRM de perfusion est une image en intensité de gris en 4 dimensions: 3 dimensions spatiales (x,y,z) pour décrire le volume du cerveau, et une dimension temporelle (t) pour décrire la dynamique de passage de l'agent de contraste. Un moyen d'analyser ces images consiste à encoder

l'environnement local de chaque voxel en signature spatio-temporelle comme décrit sur le schéma ci-dessous :

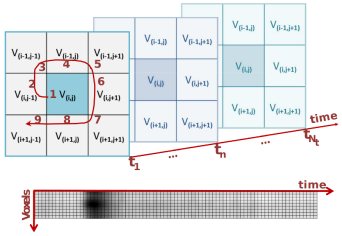

Figure 2: Exemple de signatures spatio-temporelles. Axe y: encodage du pixel et de son voisinage, Axe x: temps (*image: M. Giacalone*)

La signature spatio-temporelle est un graphique 2D traduisant une information locale 3D (x,y et t). On encode l'information contenue dans chaque voxel en déroulant les signaux temporels le long d'une dimension spatiale et en empilant les signaux des 9 voxels du voisinage (3*3) du voxel d'intérêt les uns à la suite des autres, créant ainsi une imagette de taille (9*t).

Empiriquement, on sait que les signatures spatiales des tissus lésés sont différentes des signatures des tissus sains en terme de texture (*travaux de thèse de M. Giacalone*).

2 Objectifs du projet

L'objectif de ce projet est de discriminer les pixels sains des pixels pathologiques à partir de leur signature spatio-temporelle, par une approche d'apprentissage supervisée: les SVM (Support Vector Machine). Au terme de l'apprentissage, votre modèle sera capable de prédire, pour un nouveau patch donné, sa classe de type binaire (sain ou pathologique), et plus précisément sa probabilité d'appartenir à cette classe.

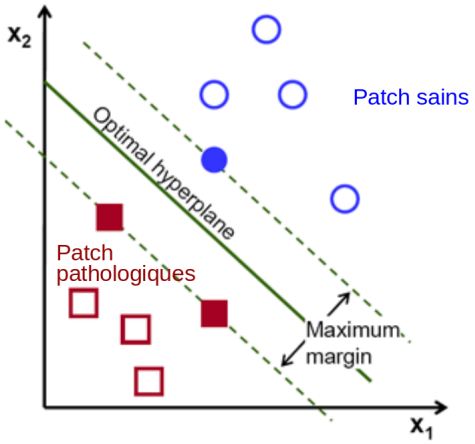

Figure 3: Schéma de répartition des patchs autour de l'hyperplan séparateur optimal après classification SVM

3 Données à disposition

Vous avez à votre disposition 2 séries de patchs générés à partir d'IRM de perfusion de patients atteints d'AVC.

Série 1: Dans cette série, certains patchs ont été créés à partir de pixels de la lésion ischémique;

d'autres à partir de pixels de tissus sains. Pour ces patchs la classification est relativement évidente.
Série 2: Dans cette série, on dispose des patchs de série 1, ainsi que d'autres patchs qui sont plus problématiques: des patchs à l'interface des tissus sains et pathologiques, ou des patchs de tissus lésés dont l'avenir est inconnu.
Tout l'enjeu est de pouvoir classer ces patchs, et de pouvoir comparer les performances de classification des 2 bases de données.

4 Étapes du projet

Pour la réalisation de ce projet, il vous est proposé de coder en python. Beaucoup de tâches qui vous sont demandées sont bien documentées sur internet et peuvent être réalisée à travers de fonctions déjà implémentées. Consultez attentivement les pages suivantes :

- $http://www.scipy-lectures.org/advanced/image\_processing/$
- $http://scikit-learn.org/stable/modules/svm.html$

4.1 Caractérisation des patchs en vecteur d'intensité de gris

Dans un premier temps, vous devez créer des vecteurs d'intensité moyenne de volume en fonction du temps.

1. Traduisez chaque patch en matrice d'intensité de gris avec le code suivant :

```
>>> from scipy import misc
>>> gray_matrix = misc.imread('patch.png')
```

2. A partir de ce chaque matrice (9*t), moyennez l'intensité de gris à chaque pas de temps pour retrouver un vecteur de taille (1*t)
3. Sauvegardez dans un fichier les 11 000 vecteurs (1*t) caractéristiques des 11 000 patchs dans un fichier.
4. Sauvegardez dans un deuxième fichier les labels associés à chaque patch (sain ou pathologique)

4.2 Entraînement du modèle

Entraînez votre modèle à partir des vecteurs d'intensité avec le code suivant :

```
 from sklearn import svm
>>> data = gray_matrix(11000,t)
>>> labels = label_vector(1,11000)
>>> clf = svm.SVC()
>>> clf.fit(data, labels)
```

N.B. : par défaut, l'hyperplan séparateur correspond à une gaussienne dans le modèle. Vous pourrez, dans la partie optionnelle, changer de type de fonction séparatrice.

4.3 Validation du modèle

1. Afin d'estimer la fiabilité de votre modèle, effectuez une validation croisée avec le code suivant:

```
scores = cross_val_score(clf, data, labels, cv=100)
score = np.mean(scrores)
```

Dans ces quelques lignes, on divise l'échantillon *data* original en 100 échantillons, puis on sélectionne un des 100 échantillons comme ensemble de validation et les 99 autres échantillons constitueront l'ensemble d'apprentissage. On calcule alors un score de performance. Puis on répète l'opération en sélectionnant un autre échantillon de validation parmi les 100 échantillons qui n'ont pas encore été utilisés pour la validation du modèle. L'opération se répète ainsi 100 fois pour qu'en fin de compte chaque sous-échantillon ait été utilisé exactement une fois comme ensemble de validation. La moyenne des 100 erreurs quadratiques moyennes est enfin calculée pour estimer l'erreur de prédiction.

2. Donnez des valeurs classiques de validation: précision, faux positifs, faux-négatifs, taux d'erreur de classification

4.4 Distance de chaque patch à l'hyperplan séparateur

1. Cherchez la fonction python dans le module SVM capable de calculer la distance de chaque point à l'hyperplan séparateur.

2. En déduire, pour chaque patch, la probabilité d'appartenir à une classe. Créez une représentation graphique pour illustrer ce résultat (une heatmap par exemple), l'objectif étant de créer une carte probabilistique d'infarctus. Notez bien qu'il y aura une heatmap par coupe axiale (coupe selon l'axe z).

4.5 Optionnel: pour aller plus loin

Vous pouvez tester d'autres fonctions séparatrices dans vos modèles d'apprentissage. Ces fonctions sont appelées des *kernels*. Les kernels les plus classiques sont les linéaires, les gaussiens et les polynomiales, mais vous pouvez préciser n'importe quelle fonction dans votre modèle.