Rapport IMA205

Matthis Maillard

April 2018

1 Introduction

Le but de ce challenge était de déterminer si des images de grains de beauté sont des mélanomes ou non. Pour effectuer cela, nous disposions d'un ensemble de 600 images pré-classifiées et la prédiction devait se faire sur 300 images dont on ne connaissait pas la classe. J'ai essayé deux approches différentes. La première a été de calculer différentes caractéristiques sur les images et ensuite d'entrainer un classifieur sur ces caractéristiques. Ma seconde approche a été d'utiliser un réseau de neurone convolutif pré-entrainé. J'ai implémenté ce projet sous python 3.6 et pour diminuer les temps de calculs, j'ai sous-echantilloné les images.

2 Extraction des caractéristiques

2.1 Asymétrie des formes

J'ai implémenté les deux features du paragraphe 2.3.1 de l'article [3]. Pour trouver les axes de symétrie de la lésion, je l'ai faite tourner de 10 degrées et j'ai calculé la mesure de symétrie décrite dans l'article. Le problème majeur de ces rotations a été que les lésions n'étaient pas centrées, donc il y a un risque que la lors de la rotation, une partie de la lésion sorte de l'image. Pour éviter cela, j'ai agrandi l'image et j'ai repositionné la lésion au centre de l'image avec la fonction de numpy pad. Ensuite, pour calculer $\Delta S1$ j'ai sélectionné la moitié supérieure de l'image, je l'ai retournée horizontalement et on fait la différence terme à terme avec la moitié du bas. Enfin, j'ai calculé la somme de la valeur absolue des termes de la matrice obtnenue. De la même manière, j'ai calculé $\Delta S2$ en sélectionnant la moitié droite de l'image. Le temps de calcul de l'ensemble de cette feature pour les 900 images est relativement long : environ 28 minutes.

2.2 Asymétrie des couleurs

Cette caractéristique est également tirée de l'article [3], paragraphe 2.3.2. Elle consiste à estimer la distribution de probabilités des pixels sur la lésion avec un noyau gaussien. J'ai utilisé la fonction gaussian_kde de scipy.stats. Pour ce calcul, il fallait également faire tourner l'image, j'ai donc procédé de la même

manière que précédemment. Pour obtenir $\Delta C1$ on calcule la différence entre la distribution de la moitiée supérieur de la lésion et la moitiée inféieur. Le temps de calcul est très très long : plus de deux heures.

2.3 Asymétrie couleur-forme

La mesure d'asymétrie couleur-forme provient du paragraphe 2.3.3 de l'article [3]. Il s'agit de mesurer la distance entre le centre de masse de la lésion et les centres de masse successifs de la lésion seuillée pour tous les percentiles [0.1, 0.2, ..., 0.9]. Les features que l'on retient son la moyenne et la variance du vecteur obtenu.

2.4 Géométrie

Ces features, sont issues du paragraphe 2.3.8. Elles consistent à seuiller l'image aux valeurs des percentiles 0.25, 0.5 et 0.75 et à calculer le nombre de composantes connectées des images obtenues. J'ai utilisé notamment la fonction ConnectedComponents d'openCV, qui retourne le nombre de composantes connexes.

2.5 Moyenne, variance, percentiles des valeurs HSV

J'ai calculé ces caractéristiques d'après l'article [2], mais je les ai légèrement modifiées. Je n'ai pas choisit le minimum et le maximum des valeurs des canaux H et V mais les valeurs des percentiles à 5% et à 95% car le maximum et le minimum pourraient êtres liées à du bruit ou des erreurs. Pour normaliser les valeurs du canal H, on lui soustrait la valeur qui a la plus grande probabilité, c'est à dire celle qui est le plus répétée sur le canal H. Pour le canal V, on lui soustrait la moyenne des valeurs qui sont en dehors du masque, c'est à dire la peau.

2.6 Moyenne, variance, percentiles des valeurs RGB

Il s'agit de la même méthode qu'au dessus mais avec les canaux RGB de l'image. On ne normalise pas les canaux.

3 Preprocessing

3.1 Augmentation des données

L'ensemble d'entrainement est constitué de 115 mélanomes et de 485 lésions bénignes. Or, pour entrainer de manière plus précise le classifieur, il faut que les classes soient en proportion égale. Pour remédier à cela, j'ai utilisé la fonction de imblearn.oversampling SMOTE qui permet de surréchantilloner l'ensemble d'entrainement pour avoir des classes en proportion égale. La taille de la l'ensemble d'entrainement est alors de 970 échantillons.

3.2 Standardisation

De plus, les features ont des valeurs très éloignées les unes des autres. Par exemple, le nombre de composantes connectées peut aller jusqu'à plus de 100 alors que les features d'asymétries des formes ne dépassent pas 1. J'ai donc centré et réduit les données en utilisant la fonction StandardScaler. Centrer et réduire est important, notamment dans le cas des svm pour éviter qu'une feature ne prenne plus d'importance que d'autres.

4 Classification

J'ai testé plusieurs classifieurs : SVM, QDA, LDA, Random Forest et Decision Tree. Pour évaluer leur précison, j'ai procédé par validation croisée. C'est également par validation croisée que j'ai choisi les hyperparamètres des classifieurs. J'ai utilisé la fonction GridSearchCV de Scickit-learn pour faire cela. La fonction de score est le coefficient de corrélation de Matthews, comme cela est évalué lorsque l'on soumet le fichier sur Kaggle.com. Pour la SVM, les paramètre à fixer sont le noyau, le coefficient de pénalisation, et dans le cas du noyau gaussien, le coefficient gamma. Pour la QDA et la LDA, je n'ai pas donné de paramètre à reglé car ils auraient une influence trop faible sur le score. Dans le cas de Random Forest, les paramètres sont la profondeur maximale des arbres et le nombre d'arbres. Enfin, pour les arbres de décision, le seul paramètre à regler que j'ai choisi est la profondeur maximale des arbres.
Les scores obtenus sont donnés dans le tableau ci-contre :

Estimateur	Score
SVM	0.821419398242
Random Forest	0.765829196
Decision Tree	0.609455596214
QDA	0.536026074001
LDA	0.463459616852

On en conclut que le meilleur estimateur semble être SVM. Les paramètre obtenus sont un noyau gaussien, C = 100 et gamma = 0.1. Random Forest est moins adapté à ce problème car nous avons trop peu de données. En effet, les méthodes d'ensemble telles que random forest ont un faible biais mais une grosse variance. Ils deviennent donc de plus en plus précis lorsque la taille de l'ensemble de test augmente.
Les arbres de décision sont moins bon que svm car ils font plus facilement et plus rapidement de l'overfitting. En pratique, l'overfitting peut être réduit en utilisant les méthodes d'ensemble mais comme nous l'avons vu précédement, il n'est pas très efficace de les appliquer ici.
En calculant la prediction sur le fichier test, on obtient une erreur de 0.23431, ce qui n'est pas satisfaisant.

On peut chercher à réduire la dimension du training set en utilisant la PCA.

On va encore procéder par validation croisée avec la fonction GridSearchCV pour calculer le score des prédictions avec différentes valeurs de n_components. Ce paramètre indique de quelle taille doit être le training set retourné par la PCA. On va utiliser un Pipeline. Cela permet de calculer le score moyen de la validation croisée en appliquant la pca avec ses différents paramètres et avec un estimateur SVM.
On obtient que pour n_components = 23, l'erreur de validation croissée est mieux qu'avec les 29 features. En revanche l'erreur est très légèrement supérieure : 0.821515933873 contre 0.821419398242 auparavant. Lorsque l'on applique la prédiction sur l'ensemble de test, on remarque que le résultat est exactement le même. Dans ce cas, la réduction de la dimension n'a pas été satisfaisante. Cela est probablement du au fait que le nombre de caractéristiques (29) n'est pas très élevé.

Le choix final de classifieur pour ces features est donc SVM avec un noyau gaussien, C = 100 et gamma = 0.1.

5 Autre approche : réseaux de neurones

Dans l'article [1], il est question de détecter les mélanomes à l'aide de réseaux de neurones et plus particulièrement avec le réseaux pré-entrainé Inception-V3. J'ai utilisé l'architecture et les poids de Inception-V3 pour calculer 2048 features. Pour faire cela, j'ai considéré la dernière fully-connected layer comme un vecteur de features. Il y a 2048 éléments donc pour chaque images, on calcule 2048 features. La taille d'entrée de mes images est 149 x 149. J'ai utilisé la fonction resize d'opencv pour changer la taille des images d'origine. Les entrées sont les images des lésions auxquelles j'ai appliqué le masque de segmentation. J'ai utilisé la fonction de cet article [4]. Une fois que j'ai calculé les features, je classifie les lésions avec SVM. J'utilise un Pipeline pour faire une validation croisée tout en trouvant les meilleurs hyperparamètres de la SVM et de la PCA. On obtient que la PCA garde 1000 features et que la SVM avec un noyau gaussien a pour paramètres C=10 et gamma=0.0001.
Le score final obtenu est de 0.37007, ce qui est une nette amélioration du score obtenu avec les features précédentes.

5.1 Conclusion

Dans ce projet, j'ai pu aborder plusieurs techniques de classification. Celle qui semble être la plus prometteuse pour la classification de mélanome est l'utilisation d'un réseau de neurone convolutif. C'est avec cette méthode que j'ai obtenu le meilleur score. On pourrait améliorer ce que j'ai fait en ré-entrainant le réseau de neurone, ce qui permettrait de modifier légèrement les poids du réseau afin qu'il soit plus adapté au problème de classification de mélanome. On aurait également pu conbiner les deux méthodes en ajoutant les features calculées dans la première partie à celles calculées par le réseau de neurones.

References

[1] Roberto A. Novoa Justin Ko Susan M. Swetter Helen M. Blau Sebastian Thrun Andre Esteva, Brett Kuprel. Dermatologist-level classification of skin cancer with deep neural networks. 2017.

[2] Reinhard Röhrer Ernst Wildling Michael Binder Harald Ganster, Axel Pinz and Harald Kittler. Automated melanoma recognition. 2001.

[3] Kevin Thon Marc Geilhufe Kristian Hindberg Herbert Kirchesch Kajsa Møllersen Jörn Schulz Stein Olav Skrøvseth Fred Godtliebsen Maciel Zortea, Thomas R. Schopf. Performance of a dermoscopy-based computer vision system for the diagnosis of pigmented skin lesions compared with visual evaluation by experienced dermatologists. 2014.

[4] May Yeung. Using tensorflow and support vector machine to create an image classifications engine. 2016.