TELECOM Paris
Rapport de projet
Classification d'images de lésions de peau

Joseph Maillard,
IMA 205, 2023-2024

1 Introduction

Les mélanomes sont l'une des causes principales de cancer auprès des populations europpéennes, et il est parfois difficile de déterminer si une lésion de peau est un mélanome ou un simple grain de beauté.

Récemment énormement de travaux ont été dirigé dans l'objectif de réaliser des classifications automatiques des lésions de peau dans un but d'aide à la décision auprès des dermatologues. Ces techniques s'appuient sur l'utilisation de méthode de Machine Learning.

L'objectif de ce projet était de construire et de tester quelques modèles de classification en ce basant sur la base de donnée ISIC. Le code est divisé en quatre Jupiter notebook :

— create_csv.ipynb regroupe toutes les fonctions de preprocessing et permet de créer un fichier csv contenant toutes les features extraites d'un dossier d'image à partir du chemin relatif vers ce dossier.

— SVM.ipynb entraine un svm à partir du fichier csv des features pour le dataset d'entrainement (Train) ainsi que les classes associés dans metadataTrain.csv, pour ensuite effectuer la classification étant donné en entrée le fichier de features du dataset de test et sauvegarde la classification dans un fichier csv.

— random_forest.ipynb effectue la même chose cette fois-ci en utilisant un random forest.

— ResNet.ipynb importe un réseau pré-entrainer ResNet 50 et le fine-tune pour l'adapter à la classification recherchée.

Un autre notebook, segmentation_clean.ipynb permet d'évaluer la qualité des ségmentations en calculant un histogramme des coefficients dice.

2 Preprocessing

2.1 Ségmentation

Pour extraire les features nécessaires à la classification des images, il faut dans un premier temps trouver une méthode de segmentation automatique de lésions de peau, étant donné que toute la base de donnée n'a pas été ségmenté.

Je me sert ensuite des ségmentations connues pour évaluer la robustesse de la méthode en calculant un histogramme des coefficients dice entre les ségmentations effectuées et celles connues.

Pour ne pas fosser les ségmentations il est nécessaire de retirer des images les pixels de cheveux, ainsi que de retirer les vignettes noires circulaires si il y en a. Cela est réalisé par les fonctions remove_hair, qui trouve les cheveux d'une image à l'aide d'opérations morphologiques, puis les efface par inpainting, et crop_image qui analyse les pixels de la diagonale pour recentrer l'image.

J'ai trouvé une méthode de ségmentation (voire Références) qui s'appuie sur le fait que les lésions ont généralement des valeurs de saturation dans l'espace HSV plus élevées que la peau. Après avoir passé l'image dans l'espace HSV et d'avoir appliqué un filtre gaussien, la méthode Otsu est utilisé pour trouvé un seuillage automatique en fonction de la forme de l'histogramme de la saturation. Après le seuillage, une ouverture est appliquée au mask de ségmentation trouvé pour arrondir les bords et supprimer les petites structures. Les différentes zones trouvées (non-connexe entre elles) sont évaluées selon leur distance au centre de l'image et on ne conserve que la zone la plus proche.

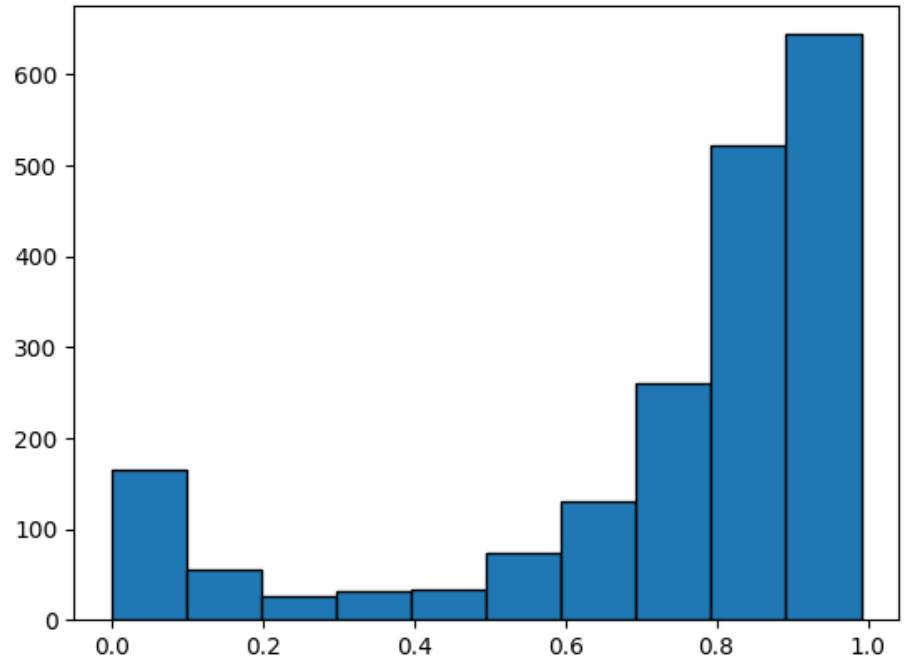

FIGURE 1 – Histogramme des coefficients dice sur les ségmentations connues

Après avoir remodifié cette fonction (en diminuant la taille des objets structurants pour les érosion et dilatations), j'ai obtenue un dice moyen de 0.725. Cela montre que cette méthode permet d'obtenir un bonne ségmentation sur l'ensamble des images, avec dans une grande majorité des cas un dice supérieur à 0,7. En supposant que l'ensamble des images dont on connait les masks est repésentatif de l'entièreté du dataset, cette méthode de ségmentation paraît relativement robuste.

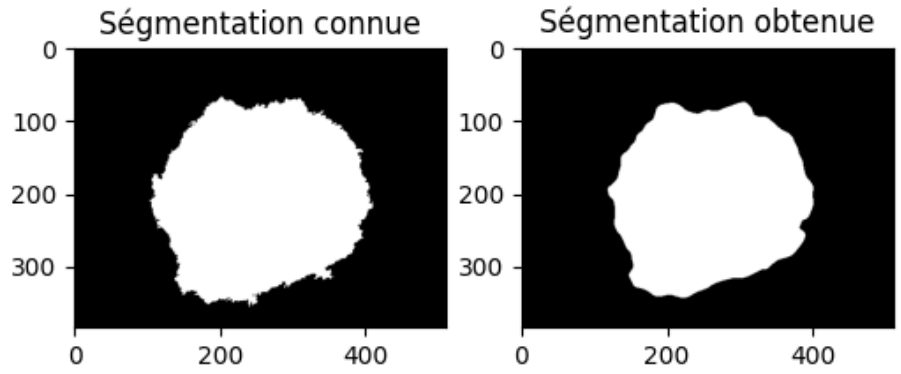

FIGURE 2 – Comparaison entre une ségmentation connue et celle obtenue

On remarque toutefois que pour certaines images, le dice est inférieur à 0,1. Ce qui signifie que la ségmentation obtenue ne se superpose pratiquement pas au mask connu. C'est potentiellement le cas pour les images où la peau a une teinte plus saturée que la lésion.

2.2 Extraction des features

Après avoir otenue le mask de ségmentation des lésions (si on a accès au mask "ground truth" alors on utilise celui-ci) on peut extraire différentes features des lésions. À noter que je n'ai utilisé que des features extraites de la partie lésion, et non pas de la peau.

Dans la littérature, la plupart des méthodes proposées se basent sur les features "ABCD" :

- L'indice d'assymétrie
- L'éccentricité
- La compacité, calculé comme le rapport du périmétre au carré sur l'aire
- Les variations de couleurs, représentées par les rapports d'écart-types de chaque canaux RGB sur le maximum de ces canaux.
- Le diamètre
- Les textures : ces features sont extraites de la matrice de co-occurence (Gray Level Co-occurrence Matrix, GLCM), elle représente la corrélation, l'homogénéité, l'énergie et le contraste.

3 Algorithmes de classification

3.1 Random forest

J'ai d'abord voulue utiliser l'algorithme du random forest pour effectuer la classification à partir des features. J'ai fixé le nombre d'arbre à 100, sans contraintes de nombre minimum d'échantillon par noeuds ou par branches. J'ai obtenue un score privé de 0,304.

J'ai ensuite décidé de fixer un nombre d'échantillons minimum par split de 20, la score a chuté drastisquement, passant à 0,131.

J'ai ensuite décidé de fixer un nombre minimum d'échantillon par branche de 3, j'ai obtenu un score de 0,295.

Normalement le fait de fixer un nombre d'échantillon minimum par noeud ou par branche est supposé stabiliser l'algorithme, en le rendant moins sensible au bootstrapping et à l'overfitting, toutefois ici l'algorithme senble apprendre moins bien des données. Le test avec un nombre d'échantillons minimum par split de 20 est clairement un cas d'underfitting, l'algorithme n'est plus capable de saisir la complexité des données.

3.2 SVM

J'ai ensuite décidé d'appliquer un SVM sur les features de Train. En calculant l'accuracy sur les données d'entrainement, j'ai estimé que je devais utilisé un noyau gaussien. J'ai réalisé un premier test avec la valeur de C par défaut de 1. Cela m'a donné un score de 0,330. J'ai ensuite augmenté la valeur du paramètre C à 300, ce qui m'a donné un score de 0,348.

Le paramètre C contrôle la complexité de l'algorithme, c'est un compromis entre la maximisation de la marge et la minimisation de l'erreur sur les données d'entraînement. Plus C est grand, plus la marge est réduite pour permettre une classification correcte des points d'entraînement.

Ici l'algorithme apprend mieux avec une valeur élevé de C, mais il reste assez limité.

4 Autres implémentations

Une autre implémentation possible pour obtenir une classification conciste à entrainer un réseau de neurones convolutionel à effectuer la classification. J'ai décidé de prendre un réseau de neurones déjà existant (et pré-entrainé), ResNet 50, et de le fine-tuner afin d'effectuer la classification des images.

Ce réseau est constitué de 50 couches. Il utilise la technique du Residual Blocks (voir fig 3) pour pallier au problème de vanishing gradients.

Le réseau de neurones a déjà été entrainé pour extraire 2048 features, qui permettent une classification en 1000 classes différentes.

Pour obtenir un classificateur en 8 classes différentes, on doit supprimer la dernière couche et la remplacer par une couche linéaire suivie d'un softmax pour permettre de déterminer la classe la plus probable.

Le modèle est pré-entrainé car les poids correspondent à ceux optimisé pour ImageNet, toutefois, au cours de l'entraînement j'ai décidé que l'entièrté des paramètres du réseau étaient entrainables pour que le réseau apprenne mieux des spécificités de la base de données.

Après entrainement de ce réseau sur la base de donnée (Fine-tuning), l'algorithme donnait finalement un score privé de 0,471 sur le dataset de test.

5 Interprétation des résultats

On remarque qu'un CNN dont l'architecture n'a pas été pensé pour cette tâche permet d'avoir des résultats très correctes, et même meilleur que les al-

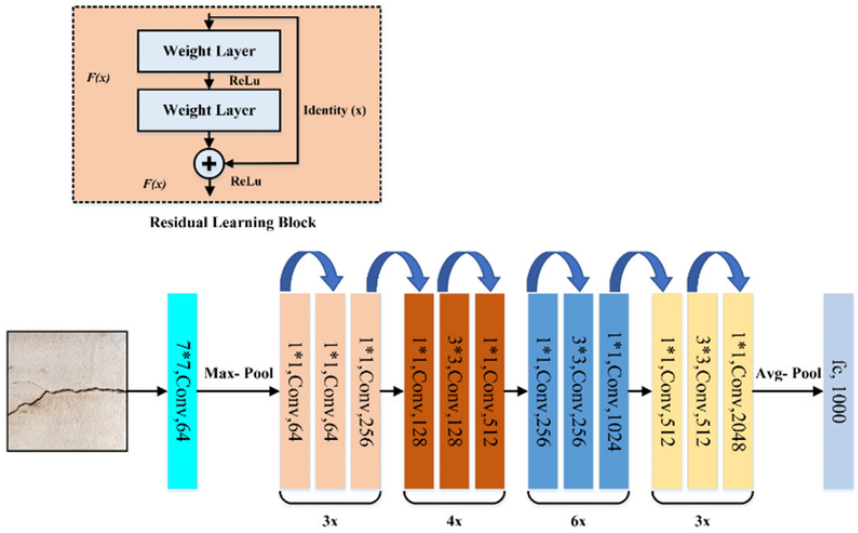

FIGURE 3 – Le schéma d'architecture de ResNet-50

gorithmes de machnie learning classiques (SVM et Random Forest). Cela est potentiellement dû au fait que d'une part ces derniers s'appuit sur des features qui ont été extraites suite à une ségmentation, et d'autres parts que ResNet 50 présente une architecture très complexe capable d'apprendre des représentations complexes.

Cela est potentiellement dû à la qualité insuffisante de la ségmentation. Comme nous l'avons vu, il est possible que le mask trouvé ne corresponde pas du tout à la zone d'intérêt. Dans ce cas les features extraites ne sont pas du tout représentatives de la lésion et peuvent fosser les algorithmes en rajoutant du bruit. Même pour des coefficients de dice "élevés" (>0.7), une petite excroissance de la ségmentation par rapport à la lésion peut facilement fosser les features (comme l'indice d'assymétrie ou la compacité).

Références

[1] Luis Cambero, *Skin lesion segmentation*, `https://github.com/hl0d0w1g/skin_lesion_segmentation`

[2] Biagio Montaruli, *Skin Lesions Classification using Computer Vision and Convolutional Neural Networks*, 9 septembre 2019, `https://github.com/biagiom/skin-lesions-classifier/tree/master`

[3] Roberta B. Oliveira, Norian Marranghello, Aledir S. Pereira, João Manuel R.S. Tavares, *A computational approach for detecting pigmented skin lesions in macroscopic images*, 1 November 2016, `https://www.sciencedirect.com/science/article/abs/pii/S0957417416302354#bib0023`

[4] Javier Velasquez, *Dullrazor algorithm*, `https://github.com/BlueDokk/Dullrazor-algorithm`