Maël Mathevet

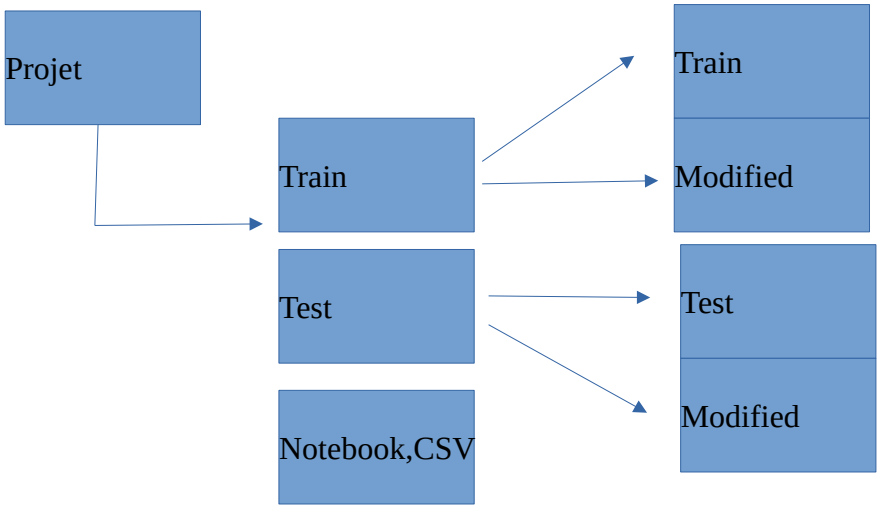
IMA 205 – Apprentissage pour l’image et la reconnaissance d’objet

Rapport de projet

Sommaire

Petit préambule :
Vous vous doutez sûrement, vu mon score de 100% sur le kaggle, que je n'ai pas réussi à battre l'état de l'art sur ISIC2019 et ai effectivement trouvé les ground truths du challenge en ligne. Cependant je ne me suis pas servi des dîtes ground truths pour l'entraînement de mes modèles et ai essayé de respecter la limite des 5 calculs d'accuracy par jour. Je ne me suis pas servi du dépôt kaggle pour avoir plus d'impact lors du dépôt de mon score final. Libre à vous de me croire, de me dire que lors de vrais challenges le nombre de soumission est limité pour éviter le reverse engineering, qui n'est d'ailleurs pas ce que j'ai fait. Je ne me destine par ailleurs pas à faire de la recherche et si je me retrouve dans le futur à devoir réaliser une tâche semblable à ce projet cela sera probablement au sein d'une entreprise et pour le compte d'un client qui aura besoin de résultats et il n'y aura alors aucune histoire de limite de soumission par jour. Vous pourriez rétorquer que si l'on triche lorsqu'il n'y a pas d'enjeu, quelle garantie y a-t-il pour que cela ne se reproduise pas dans un contexte où l'enjeu est réel et je répondrai que la pression de travailler pour quelqu'un qui a vraiment besoin des résultats de votre travail est complètement différente de celle lors d'un projet final d'un cours et je ne ferai jamais la même chose dans une situation où je sais que quelqu'un a besoin de mes compétences. J'ai simplement trouvé un moyen de m'amuser et d'amuser mes camarades, le score final que j'obtiens est assez faible par rapport aux meilleurs élèves du cours qui ont utilisés des modèles bien plus performants, avoir trouvé les ground truths ne m'a confié aucun avantage sur les autres élèves pour ce qui est du résultat final, j'aurais écrit exactement le même programme ligne pour ligne si je ne les avait pas eu.

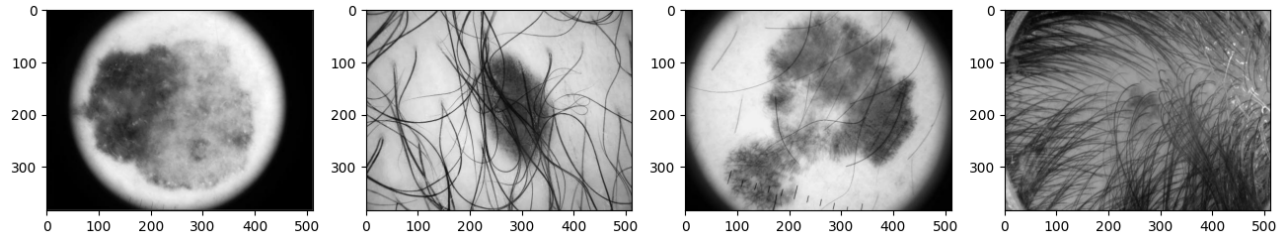
Second point important : l'architecture du dossier dans lequel se trouve mon projet. Le notebook ainsi que les csv contenant les métadonnées se trouvent tous dans le même dossier. Ce dossier contient deux autres dossiers : Train et Test qui contiennent chacun un dossier Modified ainsi qu'un dossier portant le même nom qu'eux mêmes. Le dossier Modified contient toutes les images du Train/Test après préprocessing ainsi que leur masque de segmentation associé alors que le dossier Train/Test est celui que l'on pouvait trouver sur Kaggle.



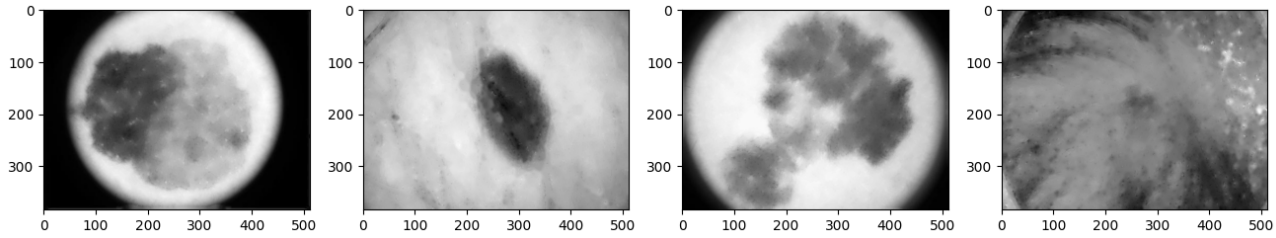
ISIC (International skin image collaboration) est une base de données d'images de lésions de peau proposant régulièrement des challenges dont l'objectif consiste à classifier au mieux un ensemble d'images parmi différentes classes (exemple : mélanomes, névus, etc). La reconnaissance d'image dans le domaine médical est extrêmement importante car elle promet une accélération du processus de diagnostic et améliorerait grandement les conditions de travail des médecins. Le challenge final d'IMA 205 portait sur les données du challenge 2019 d'ISIC composé de 25331 images. En regardant rapidement les différentes images, de nombreux problèmes apparaissent : Toutes les images ne font pas les mêmes dimensions, ne possèdent pas un masque de segmentation, certaines images présentent une zone noire sur la périphérie ou encore une règle pour pouvoir mesurer les dimensions réelles sur la photo ainsi que des stickers colorés pour cacher les lésions dans les situations où plusieurs apparaissent dans le champ de la photo. On trouve également des image avec de nombreux poils qui gênent la détection et certaines lignes du tableau des métadonnées sont vides.

I- Preproccesing :

J'ai d'abord commencé par essayer de régler le problèmes des poils et des zones noires avant de procéder à la segmentation des images ne possédant pas déjà de masque. Le second problème a été le plus simple à résoudre : j'ai simplement créé un masque contenant tous les pixels de très faible intensité auquel j'ai appliqué une opération de dilatation morphologique afin de dépasser la limite entre zone sombre extérieure et zone claire centrale puis j'ai effectué un inpainting sur l'image originale. Pour supprimer les poils sur les images j'ai d'abord essayé d'implémenter l'article 'An effective hair removal algorithm for dermoscopy images'[1] par Mohammad Taghi Bahreyni Toossi et al. Cet article expose une méthode qui consiste à appliquer un PCA sur l'image afin d'obtenir la composante de contraste maximum puis de lui appliquer une version modifiée d'un filtre de canny. Cependant certains passages de l'article manquent de clarté et je n'ai pas été en mesure de reproduire les résultats qu'il présente ce pourquoi je me suis rabattu sur une méthode plus simple et assez satisfaisante : l'algorithme Dull-razor. Cet algorithme consiste à effectuer une opération de blackhat sur l'image (différence entre fermeture de l'image et image originale) avec un noyau linéaire puis de seuiller le résultat pour obtenir un masque utilisé ensuite pour de l'inpainting.
L'algorithme donne les résultats suivants, de gauche à droite une image avec bord noirs sans poils, une image sans bords noirs avec poils, une image avec bords noirs avec poils et enfin une image de crane avec presque exclusivement des poils :

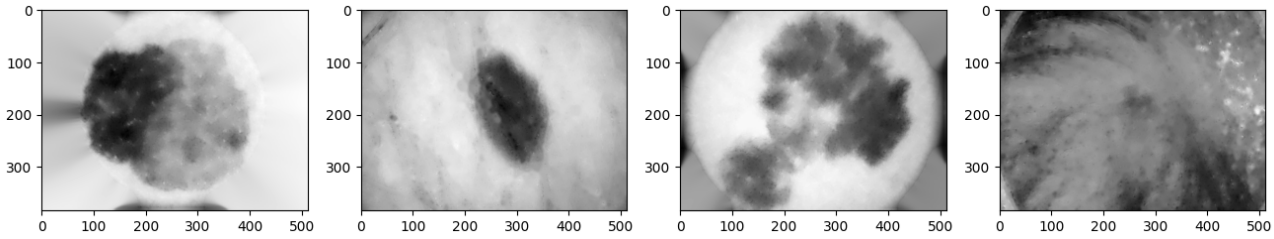


Images originales

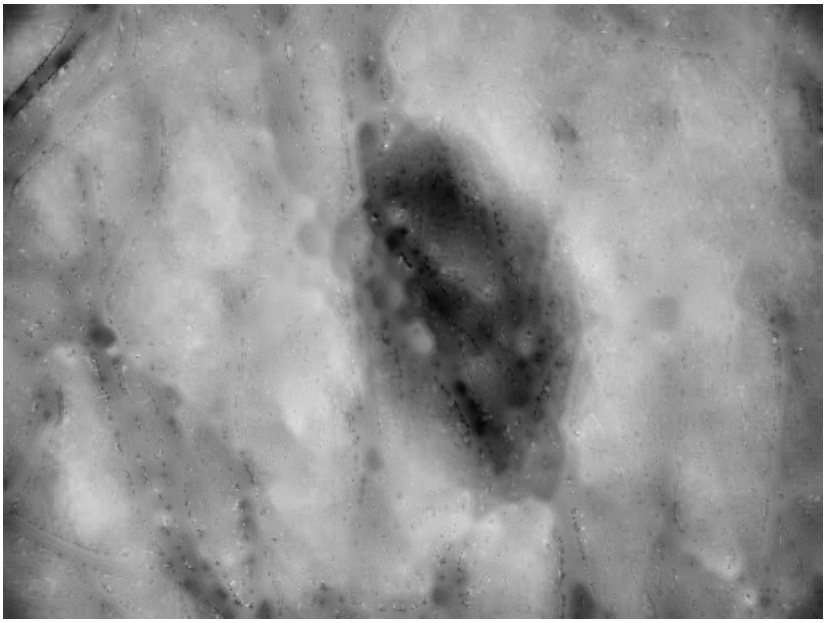


Images après dull-razor

Après avoir appliqué l’algorithme dull-razor j’ai ensuite réglé le problème des bordures noires :



Et j’ai enfin appliqué l’algorithme CLAHE (Contrast limited adaptive histogram equalization) qui consiste à effectuer découper l’image en blocs, ici de 8x8 pixels, et à égaliser les histogrammes de chacun d’eux. On obtient donc en sortie des images de cette sorte :



L’image est plus contrastée, il est possible de distinguer plus de nuances de gris au sein de la lésion ce qui devrait faciliter la segmentation.

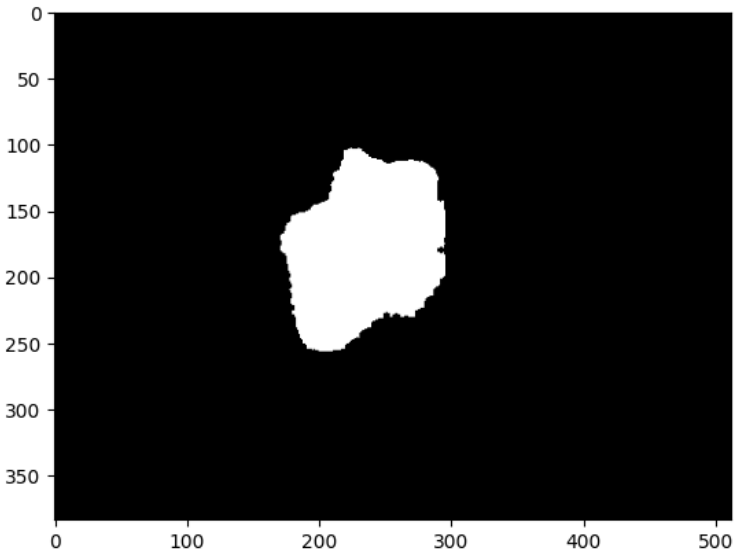
II- Segmentation :

Pour obtenir un masque de segmentation pour les images qui n'en avaient pas à l'origine, c'est à dire la plupart d'entre elles, je me suis immédiatement tourné vers le deep learning. J'ai pensé à essayer une méthode utilisant les kmeans avec entre 5 et 10 classes puis en gardant seulement un certain nombre d'entre elles mais cette idée ne me paraissait pas extraordinaire. Après un peu de recherche je suis tombé sur un article[2] présentant une architecture avec trois couches convulotionnelles suivies de trois couches convolutionnelles transposées puis une dernière convolution qui permet d'obtenir des images qui ne sont pas totalement des masques mais qui peuvent le devenir facilement avec un seuillage :

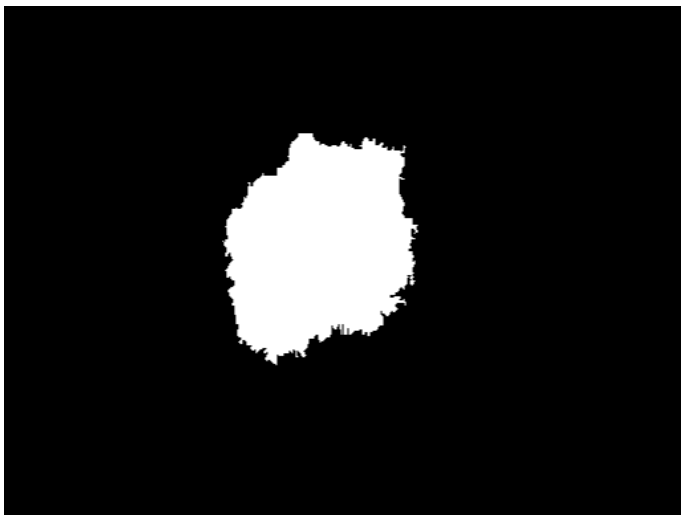
```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input (InputLayer)          [(None, 384, 512, 1)]     0

 conv2d (Conv2D)             (None, 192, 256, 16)      160

 conv2d_1 (Conv2D)           (None, 96, 128, 8)        1160

 conv2d_2 (Conv2D)           (None, 48, 64, 4)         292

 conv2d_transpose (Conv2DTr  (None, 96, 128, 4)        148
 anspose)

 conv2d_transpose_1 (Conv2D  (None, 192, 256, 8)       296
 Transpose)

 conv2d_transpose_2 (Conv2D  (None, 384, 512, 16)      1168
 Transpose)
```

J'ai entraîné ce réseau sur les images de l'ensemble d'entraînement disposant déjà d'un masque de segmentation en ramenant leur valeur entre 0 et 1 et me suis servi des images de l'ensemble de test pour déterminer le meilleur seuil pour obtenir les masques finaux.
Pour cela j'ai prédit des masques grâce à mon réseau puis en ramenant les valeurs de l'image obtenue entre 0 et 255 j'ai calculé le ratio de l'aire de la différence entre le masque réel et le masque prédit sur l'aire du masque réel pour chaque valeur de seuil de 0 à 255 et ait gardé comme seuil final la valeur causant la plus petite différence pour toutes les images. Cependant, les résultats n'étaient pas satisfaisants pour les images ayant à l'origine une région toute noire sur les bords, même après l'inpainting de la section précédente. Suite à cette observation j'ai décidé de rajouter deux étapes avant de calculer le ratio des aires : j'ai vérifié si la majorité des points non nuls du masque prédit se trouvaient sur les bords, auquel cas je multiplie maintenant l'image par un masque carré qui ne garde que le centre de l'image puis ai ajouté une bande noire autour de l'image pour éviter les problèmes causés par les objets touchant les bords de l'image et ai appliqué une méthode de détection d'objets[3] pour ne garder que le plus grand d'entre eux.

Voici un exemple de masque obtenu côte à côte avec le vrai masque :

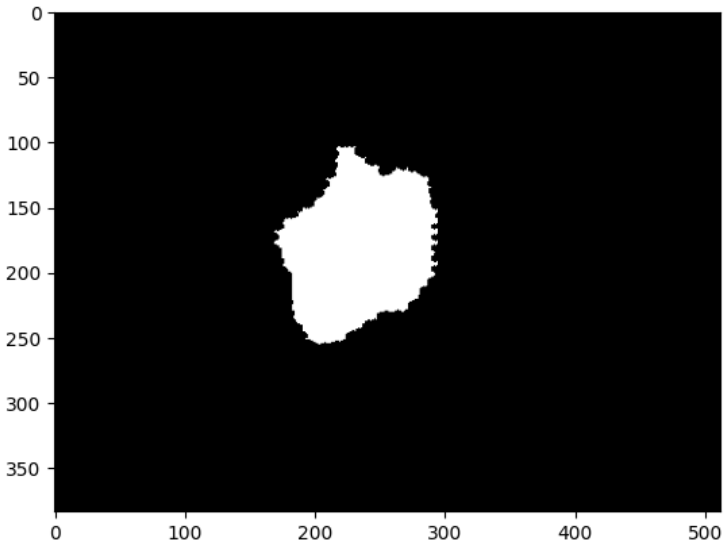


Masque prédit



Masque réel

J’ai aussi pu tester l’intérêt de l’utilisation de CLAHE en entraînant le modèle sur les images avec et sans CLAHE et l’augmentation de contraste permet effectivement un meilleur résultat, le masque prédit sans utilisation de CLAHE est plus éloigné du masque réel:



J’ai ensuite ajouté les masques déjà existant dans le même fichier puis, m’étant rendu compte que certaines images n’avaient pas de segmentation associées car aucun contour n’avait pu être trouvé j’ai vérifié que celles-ci étaient très rares (25 sur les 6333 de l’ensemble de test et 5 sur les 18998 de l’ensemble d’entraînement) et leur ai associé comme masque une image entièrement noire.

III- Remplissage des trous dans les métadonnées et calcul des features :

Comme j'ai pu le remarquer plus haut, les fichiers contenant les métadonnées des lésions étaient incomplets, des valeurs de sexes, ages et positions manquaient. J'ai donc calculé la proportion de chaque sexe et ait rempli les champs manquant en tirant au hasard, j'ai fait de même pour les positions. Pour l'âge j'ai choisi de tirer l'âge au hasard selon une loi uniforme entre l'âge minimum et l'âge maximum des patients. Une piste d'amélioration aurait été d'étudier la répartition pour obtenir plus d'informations et utiliser une loi plus proche de la réalité qu'une loi uniforme.

Pour obtenir mes différentes features j'ai d'abord convertit le sexe, l'âge et la position en un nombre réel entre 0 et 1. Le choix du nombre pour la position a été fait selon la proximité des zones sur le corps : Avec le torse à 0 j'ai choisi de mettre ensuite les côtes puis le dos, la tête, la bouche/les parties génitales, les bras, les jambes et enfin les paumes/plantes des pieds. L'idéal aurait peut être été de transformé cette feature en 8 différentes et d'avoir seulement un 1 pour chaque image mais cela me semblait multiplier le nombre de features pour un gain qui serait probablement assez faible.

Ensuite j'ai décidé quelles features extraire de mes images grâce à un article cité dans l'introduction d'un TP[4]. Les features que j'ai retenue peuvent se partager en deux catégories, d'abord les features obtenues grâce à la segmentation[5] : l'aire de la lésion, son périmètre, l'aspect ratio de sa bounding box, la distance entre son centre de gravité et celui de sa bounding box, sa compacité, sa rondeur, sa convexité et sa solidité et les features obtenues sur l'image d'origine : la valeur minimum, maximum, la moyenne et la variance des canaux H et L lors de la conversion dans l'espace HSL de l'image ainsi que la différence de la moyenne des canaux R,G,B,H et L entre l'intérieur de la lésion et le reste de l'image. J'ai ensuite normalisé toutes les valeurs pour lesquelles cela était nécessaire entre 0 et 1. Je précise celles pour lequel cela était nécessaire car la compacité, la rondeur, la convexité et la solidité sont des grandeur qui par définition ont une valeur comprise entre 0 et 1. Cependant quelques images ont produit des valeurs supérieures à 1 pour ces grandeurs, j'ai donc imposé un maximum à 1 sans oublier de vérifier que cela ne créait pas une grandeur pour laquelle toutes les images avaient une valeur de 1, en réalité seul un très faible nombre d'entre elles dépassaient la limite.

IV- Classification :

J’ai testé différents algorithmes de classification : un LinearRegressor, un LogisticRegressor, un Decision Tree, une Random Forest et finalement un SVM. J’ai d’abord essayé des valeurs au hasard pour chacun des modèles à paramètres pour savoir quelles valeurs tester lors d’éventuelles cross-validations. J’ai ensuite fait tourner ces cross-validations (Logitcv et GridSearchCv pour Random Forest et SVM) avec pour score de validation la précision rééquilibrée par des poids inversement proportionnels à la taille de chaque classe puisque c’est cette mesure qui sert à nous évaluer.
J’ai finalement entraîné un modèle avec les paramètres optimaux obtenus par cross-validation.

Le point important lors de l’entraînement était de bien sélectionner l’option ‘balanced’ pour le champ class_weight pour chaque méthode qui le possédait. Pour les régression logistique et linéaire l’option ‘multinomial’ du champ multi_class permettait de sortir du cadre binaire. Finalement, pour les méthodes arborescentes j’ai choisi l’entropie comme critère de séparation pour maximiser la représentation de chaque classe et ne pas se retrouver avec un modèle qui place toutes les lésions dans la classe la plus représentée (qui compte pour la moitié de la base de données).

Le classement des méthodes utilisée est le suivant : en dernier la régression linéaire avec 21,9% puis la random forest avec 22,2 % qui même après une cross-validation n’a pas pu dépasser l’arbre de décision et ses 29,4% qui la suit mais précède la régression logistique avec 34,9 % et enfin le SVM qui obtient le meilleur score : 46,2 %.

[1] :https://www.researchgate.net/publication236125582_An_effective_hair_removal_algorithm_for_dermoscopy_images#:~:text=Methods%20The%20proposed%20algorithm%20includes,multE2%80%90resolution%20coherence%20transport%20inpainting.

[2] : https://towardsdatascience.com/image-segmentation-using-varieties-of-auto-encoders-in-tensorflow-manual-back-prop-with-tf-2e688f2a98f7

[3] :https://stackoverflow.com/questions/56589691/how-to-leave-only-the-largest-blob-in-an-image

[4] :https://www.researchgate.net/publication/220099992_Automated_Melanoma_Recognition

[5] :http://www.cyto.purdue.edu/cdroms/micro2/content/education/wirth10.pdf