Detecting breast cancer metastases

Clement Lelong et Thomas Duchemin

Introduction

Le but de ce data challenge est de permettre une détection automatisée de métastases dans les ganglions d'un patient ayant le cancer du sein, afin de diagnostiquer la gravité du cancer. On dispose pour cela de très grandes images (100 000 * 100 000 pixels) de ces ganglions. Le médecin doit normalement étudier cette très grande image pour trouver les métastases s'il y en a. C'est un exercice très long et des erreurs sont possibles. Une procédure automatique par machine learning serait donc très bénéfique.

Jeu de données

Pour ce data challenge, on dispose d'un jeu de données de 400 patients. 280 patients sont utilisés pour entrainer le modèle, les 120 restants servent à mesurer la performance du modèle.

Pour chaque patient, on dispose d'une très grande "slide" de 100 000 * 100 000 pixels découpée en 1000 "tiles". Chaque tile mesure 896 * 896 pixels, redimensionnée à 224 * 224 pour des raisons computationnelles. Les metastases peuvent être très localisées et peu nombreuses. Le but étant de dire si le patient a des métastases ou non, il faut donc bien parcourir chaque tile. Une seule tile tumorale implique en effet un diagnostique métastasé.

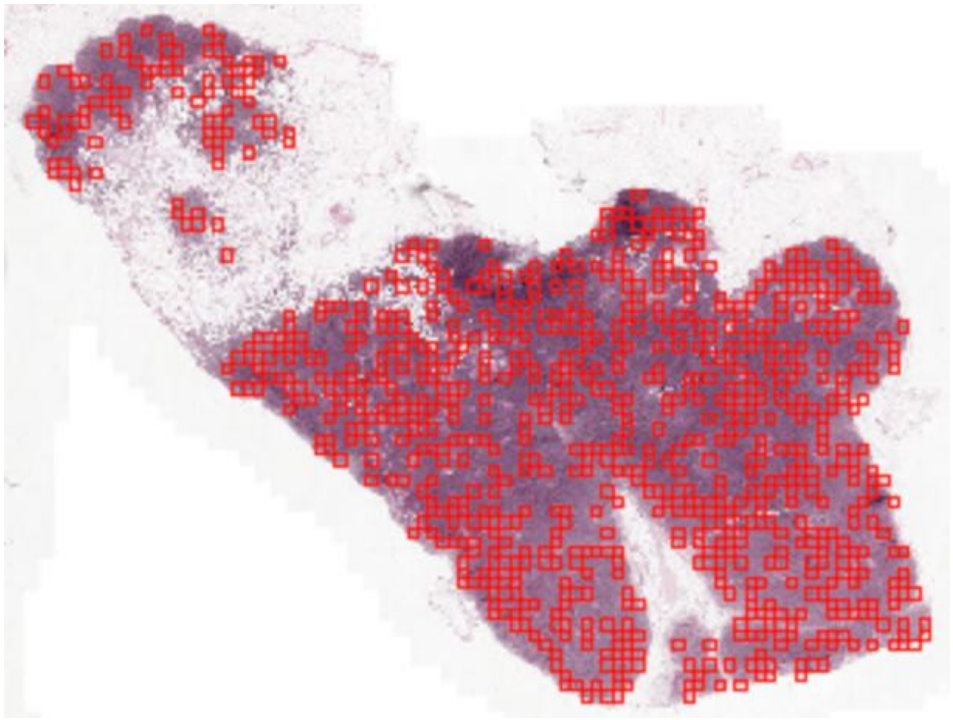

FIGURE 1 – Exemple de découpage d'une slide en différentes tiles (chaque carré rouge représente une tile)

Pour chaque tile, 2048 features ont été extraites à partir du réseau de neurones ResNet50, pré-entrainé sur une base de données d'images. On peut penser que ces features ne sont pas optimales pour étudier des images de tissus, néanmoins ces features permettent de bien mieux différencier les tiles que les images elles-mêmes. De plus, utiliser ces features plutôt que les images complètes présente un intêret évident pour le coût computationnel.

Au final, les données dont nous disposions pour ce challenge étaient :

— Les tiles pour chaque patient

— Les features correspondants à chacune de ces tiles

La difficulté de ce data challenge réside dans le fait que le problème est "weakly-supervised". En effet on dispose des labels à l'échelle de la slide, mais pas à l'échelle de la tile dans la plupart des cas. 11 slides ont cependant été annotées tile par tile. Sur ces 11 slides on sait donc pour chaque tile si elle est tumorale ou non. En résumé, les tiles annotées individuellement sont labelisées 0 ou 1 en fonction de la présence de métastase sur la tile, alors que les autres tiles sont labelisées 0 ou 1 en fonction de la présence de métastace sur la slide associée.

Objectifs

L'objectif est de renvoyer pour chaque patient de l'ensemble test, la probabilité qu'il ait des metastases.

Méthodologie et Algorithmes

Traitement des données

Avant toute chose, il nous semblait primordial de séparer les tiles annotées individuellement des tiles annotées à l'échelle des slides. En effet, il ne nous semblait pas logique d'entrainer un classifieur sur l'ensemble des données, étant donné que le problème n'est pas le même selon la méthode d'annotation des tiles.

Nous avons donc dans un premier temps décidé de travailler avec les tiles annotées individuellement. Etant donné que seules 11 slides ont été annotées ainsi, nous avons pu séparer manuellement ces 11 slides des autres, et les placer dans un dossier différent. Nous avons fait le choix de travailler sur les tiles annotées individuellement car il nous semblait qu'un classifieur entrainé sur ces tiles serait plus précis, et pourrait ensuite être utilisé sur n'importe quelle tile individuellement. De plus, même si ce choix a limité le nombre de données à notre disposition, nous disposions tout de même de 11 slides, soit 10 124 tiles.

Importation des features

Pour les features, nous disposions, pour chaque patient, d'une matrice de taille (nombre de tiles * 2051). Les 2051 colonnes correspondent aux 2048 features extraites par le réseau ResNet50, au niveau de zoom et aux coordonnées de la tile.

Notre choix a été de concaténer toutes ces matrices dans un seul array X de dimension (10 124 * 2051). Dans cet array, les tiles sont ordonnées par ID du patient,puis par position des tiles.

Importation des labels

Les labels des tiles annotées individuellement sont fournies à part, dans un fichier csv. Ce fichier comporte uniquement 2 colonnes : "TileName" et "Target". La colonne "TileName" correspond au nom de la tile, et la colonne "Target" correspond au label associé (0 ou 1 en fonction de la présence d'une tumeur).

TileName	Target
ID_387_annotated_tile_0_15_69_30.jpg	0
ID_387_annotated_tile_1_15_23_53.jpg	0
ID_387_annotated_tile_2_15_58_20.jpg	0
ID_387_annotated_tile_3_15_67_12.jpg	0
ID_387_annotated_tile_4_15_57_20.jpg	0
ID_387_annotated_tile_5_15_25_49.jpg	0
ID_387_annotated_tile_6_15_66_41.jpg	0
ID_387_annotated_tile_7_15_57_10.jpg	0
ID_387_annotated_tile_8_15_30_74.jpg	0
ID_387_annotated_tile_9_15_35_41.jpg	0
ID_387_annotated_tile_10_15_58_19.jpg	0
ID_387_annotated_tile_11_15_61_13.jpg	0
ID_387_annotated_tile_12_15_35_65.jpg	0
ID_387_annotated_tile_13_15_85_28.jpg	0
ID_387_annotated_tile_14_15_72_13.jpg	0
ID_387_annotated_tile_15_15_53_48.jpg	0
ID_387_annotated_tile_16_15_66_31.jpg	0
ID_387_annotated_tile_17_15_81_19.jpg	0
ID_387_annotated_tile_18_15_52_56.jpg	0
ID_387_annotated_tile_19_15_88_33.jpg	0
ID_387_annotated_tile_20_15_30_67.jpg	0
ID_387_annotated_tile_21_15_56_21.jpg	0
ID_387_annotated_tile_22_15_73_38.jpg	0
ID_387_annotated_tile_23_15_71_8.jpg	0
ID_387_annotated_tile_24_15_25_66.jpg	0

FIGURE 2 – Fichier contenant les labels des tiles

La difficulté liée à l'importation des labels était que l'ordre des tiles dans ce fichier était différent de l'ordre des tiles dans notre matrice X. Pour rappel, les tiles dans la matrice X sont ordonées par ordre croissant d'ID du patient, puis par ordre croissant de numéro de tiles.

Pour réordonner de manière similaire les labels, nous avons d'abord importé le fichier csv des labels dans un Dataframe Pandas. Nous avons ensuite récupéré, à l'aide des informations contenues dans la colonne "TileName", l'ID du patient ainsi que le numéro de chaque tile. Nous avons ensuite ordonné le Dataframe selon l'ID du patient, puis selon le numéro de la tile. On retrouve ainsi le même ordre que dans la matrice X.

	TileName	Target	SlideNumber	TileNumber
9258	ID_035_annotated_tile_0_16_56_117.jpg	1	35	0
9259	ID_035_annotated_tile_1_16_47_136.jpg	0	35	1
9260	ID_035_annotated_tile_2_16_38_117.jpg	0	35	2
9261	ID_035_annotated_tile_3_16_40_128.jpg	0	35	3
9262	ID_035_annotated_tile_4_16_47_117.jpg	0	35	4
9263	ID_035_annotated_tile_5_16_44_122.jpg	0	35	5
9264	ID_035_annotated_tile_6_16_40_130.jpg	0	35	6
9265	ID_035_annotated_tile_7_16_33_119.jpg	0	35	7
9266	ID_035_annotated_tile_8_16_37_128.jpg	0	35	8
9267	ID_035_annotated_tile_9_16_69_157.jpg	0	35	9

FIGURE 3 – Dataframe ordonné selon l'ID du patient et le numéro de tiles

Nous avons ensuite créé un vecteur "label" contenant uniquement la colonne "Target" du Dataframe, et que nous avons utilisé pour l'entrainement.

Importation des données de test

Nous avons ensuite importé les données de test, c'est à dire les données sur lesquelles le classifieur sera évalué au final. Comme pour les features de train, ces données sont rangées dans des matrices de taille (nombre de tiles * 2051). On importe chacune de ces matrices dans un array correspondant. On dispose donc au final de 120 arrays (un pour chaque patient dans le test) de taille (nombre de tiles * 2051).

On récupère aussi dans une liste les ID de chaque patient, qui seront utiles dans la création du fichier output final.

Entrainement du classifieur et résultats

Déséquilibre des classes

Une des caractéristiques principale du problème est le fait que la répartition des 2 classes soit fortement déséquilibrée. En effet, plus de 99% des tiles ne comprennent pas de tumeur, et ont donc un label associé valant 0. A cause de ce déséquilibre, utiliser uniquement le score du classifieur pour juger de sa qualité est insuffisant. Par exemple, si on imagine un classifieur renvoyant toujours 0, alors ce classifieur aura une précision moyenne de 99% sur le jeu de données, malgrès le fait qu'il soit incapable de détecter une tumeur.

Ainsi, pour juger la qualité de nos modèles, nous avons utilisé en plus du score 3 autres grandeurs : la précision, le recall et le F1 score. Ces grandeurs sont définies ainsi :

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1Score = 2\frac{Precision * Recall}{Precision + Recall}$$

Où TP est le nombre de Vrai Positif, FP le nombre de Faux Positif et FN le nombre de Faux Négatifs. L'objectif va donc être d'avoir un classifieur avec un F1 Score le plus grand possible (pour maximiser à la fois la précision et le recall).

Si l'on reprend l'exemple précédent d'un classifieur absurde renvoyant toujours 0, alors on voit que le F1 Score d'un tel classifieur serait de 0, étant donné qu'on aurait TP=0. Le F1 Score parait donc être une métrique adapté à notre problème.

A noter que pour un classifieur donné, avec des paramètres fixés, qui renvoie la probabilité qu'une tumeur soit présente dans la tile, alors la Precision et le Recall ne dépendent que du threshold que l'on fixe pour la décision. En d'autres termes,pour un classifieur donné le F1 score ne va dépendre que de la valeur de la probabilité limite pour laquelle on considère qu'il y a présence d'une tumeur.

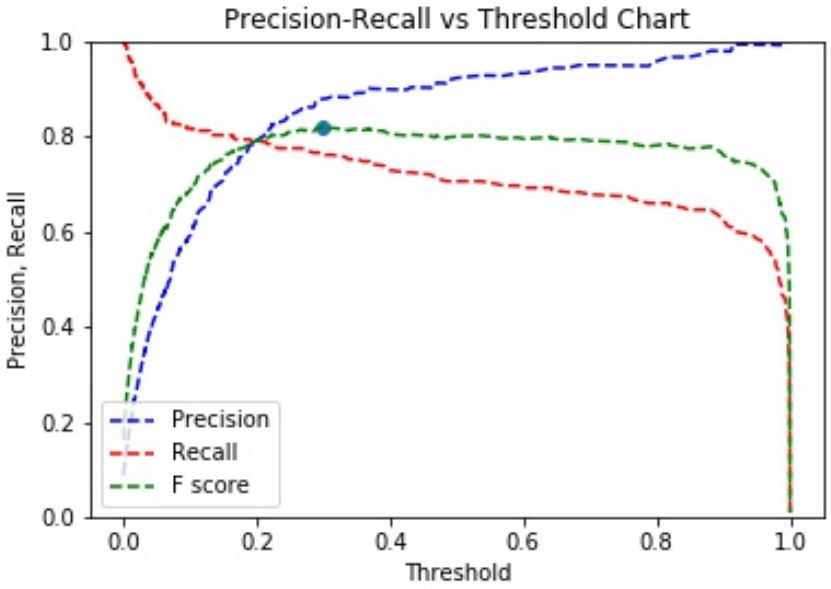

FIGURE 4 – Evolution de la précision, du recall et du f1 score en fonction du threshold pour une classifieur régression linéaire avec C fixé

Entrainement

L'objectif du classifieur est pour chaque tile de donner une probabilité d'être tumorale ou non. Sur l'ensemble des tiles annotées individuellement on prend donc un ensemble d'apprentissage et un ensemble de test pour choisir le meilleur classifieur et ses paramètres. Nous avons testé la regression logistique et le support vector machine. Pour choisir le paramètre C de la régression logistique puis du svm, on teste plusieurs valeurs en prenant à chaque fois le threshold renvoyant le meilleur F1 Score.

La méthode employée pour chaque valeur de C est donc :
- fit une regréssion logisitique sur l'ensemble train
- choix du threshold maximisant le F1 score
- prediction sur l'ensemble test en utilisant le threshold choisi
- calcul du score sur le test

En se référant au notebook, on montre que la regression logistique obtient les meilleurs résultats avec C = 0.045. On a donc un estimateur capable d'estimer la probabilité qu'une tile présente

des métastases. Il reste à évaluer la probabilité qu'un patient ait des metastases sachant les probabilités de toutes ses tiles.

Nous avons aussi essayé d'utiliser les images des tiles annotées directement. En effet les features sont renvoyées par ResNet50 pré-entrainé sur des images quelconques. Il nous semblait donc qu'essayer de classifier les tiles par un réseau de neurone entrainé sur les tiles exclusivement pourrait donnait de meilleures performances. Cependant en pratique le réseau n'a jamais réussi à apprendre à differencier une tile tumorée d'une tile saine. Nous sommes donc resté sur les features du ResNet50.

Calcul de l'output

L'output devant être soumis sur le site est un vecteur de longueur 120, dont l'élément i est la probabilité que la patient i ait une tumeur.

Il nous faut donc trouver une relation entre la probabilité qu'un patient ait une tumeur et les probabilités qu'une tumeur soit présente dans chacune de ses tiles (information dont l'on dispose). Notre premier raisonnement fut le suivant : on note Y la variable aléatoire correspondant au nombre de tumeurs d'un patient, et X_k le nombre de tumeurs dans la tile k du patient.

On a donc :

$$\begin{aligned} P(\textit{Le patient a une tumeur}) &= P(Y \neq 0) \\ &= 1 - P(Y = 0) \\ &= 1 - P((X_1 = 0) \cap ... \cap (X_n = 0)) \\ &= 1 - \prod_{i=1}^{n} P(X_i = 0) \end{aligned}$$

en considérant les tiles indépendantes

$$= 1 - \prod_{i=1}^{n} (1 - p_i)$$

p_i étant la probabilité calculée par le classifieur.

Cependant, cette formule ne donne pas de résultats concluants, à cause du trop grand nombre de tiles : le produit des $(1 - p_i)$ est en pratique toujours égal à 1 ou à 0.

La difficulté du calcul de l'output, c'est à dire la probabilité qu'un patient ait des métastases ou non, réside dans le fait que si une seule tile comporte des métastases alors le patient est considéré comme métastasé. Vu notre méthodologie cela pose problème car si l'on se trompe sur une tile et on la considère comme tumorale par erreur, le patient est automatiquement considéré comme métastasé. On a donc testé plusieurs façons de calculer la probabilité finale :

- le maximum des probabilités obtenues pour les tiles. Dans ce cas une erreur sur des tiles non tumorales peuvent impliquer un mauvais verdict sur le patient
- la 5ème plus grande probabilité sur les tiles. On se laisse ainsi une marge d'erreur sur 4 tiles. Cependant si un patient métastasé ne possède qu'une tile tumorée, on risque de la rater et donner un mauvais verdict.
- prendre la moyenne des 5 plus grandes probabilités. Ainsi on se laisse une certaine marge d'erreur tout en prenant en compte la possibilité d'avoir peu de tiles tumorées.

Or, la première méthode a pour inconvénient d'être fortement sensible aux faux positifs : en effet, si un patient ne possède qu'une seule tile pour laquelle les chances d'avoir une tumeur sont

grandes, alors on est sans doute en présence d'un faux positif. En effet, si un patient possède des tumeurs, alors il y a de fortes chances que celles ci soient présentes sur plusieurs tiles.

Nous avons donc finalement choisi d'utiliser la troisième méthode, après avoir comparé ses résultats avec la deuxième.

Résultats

Les 2 méthodes que nous avons testé sont donc une régression logistique ainsi qu'un support vecteur machine. Pour ces 2 méthodes, nous avons utilisé le f1 score pour déterminer les meilleurs paramètres à utiliser.

Ensuite, il nous a fallu trouver la meilleur façon de calculer la probabilité qu'un patient ait des métastases, sachant la probabilité que chacune de ses tiles présente une métastase. Pour cela, nous avons tester 3 méthodes différentes, avant de finalement nous arrêter sur la moyenne des 5 plus grandes probabilités des tiles.

Au final, le meilleur résultat que nous avons atteint sur le test est un score AUC de 0.8090, en utilisant une regression logistique de paramètre C=0.045. Ce score est supérieur au benchmark de 0.1.

Le support vector machine quant à lui ne nous à pas permis d'obtenir des scores satisfaisants, le meilleur résultat obtenu avec cette méthode étant de 0.708.

Conclusion

La première difficulté que nous avons rencontré durant ce data challenge fut le choix des données à utiliser. En effet, nous disposions de données labelisées de deux manières différentes, et il nous semblait impossible d'utiliser les deux types de labelisation en même temps.

Une autre difficulté que nous avons rencontré est le fait que les classes soient à ce point déséquilibrées, ce qui a rendu la comparaison d'algorithme assez délicate, et qui nous a forcé à utiliser des métriques comme le f1 score.

Enfin, la difficulté principale était sans doute le fait que l'output à fournir soit finalement assez différent des grandeurs que prédisait notre algorithme. En effet, comme nous ne disposions pas de formule exacte pour calculer la probabilité sur une slide complète à partir de la probabilité sur chaque tile, cela rajoutait une variable au problème. Nous ne pouvions pas vraiment savoir si les erreurs commises sur l'échantillon de test étaient dues à notre classifieur qui n'était pas assez performant, ou à la formule que nous utilisions pour passer au probabilité sur les slides.