Comment concourir à un challenge biomédical basé sur de l'IA

Nicolas Boutry

Mai 2022

1 iSeg2017

`http://iseg2017.web.unc.edu/`

1.1 Présentation de la méthodologie dans le cas d'un réseau de segmentation en cascade

→ voir `https://www.youtube.com/watch?v=hd4gU8Czr40`

+ Créez vous pour commencer un compte google colab, on passera sur super calculateur après.

1.2 A votre tour !

Pour télécharger les données: `https://iseg2017.web.unc.edu/download/`

1. Installer python entre 3.5 et 3.8 dans un environnement virtuel, puis à l'intérieur, ajouter matplotlib, tensorflow 2.3, numpy, jupyter (optionel), et nibabel (pour la lecture des images biomédicales).

2. prendre connaissance des données à traiter: quelle vérité terrain veut-on réussir a reproduire via nos prédictions ? (observer les images données, leurs dimensions, leurs labels ...) Et surtout: doit-on traiter le probleme en 2D ou a-t-on assez de puissance pour traiter la 3D ? Mais inversément, a-t-on assez de données pour apprendre sur de la 3D ou doit-on simuler une grande quantité de données en découpant les données 3D en slices 2D ?

3. en fonction du nombre de patients dont on possède la vérité terrain, réfléchir à une répartition données d'apprentissage (train + validation) et de test.

4. réfléchir à une normalisation des inputs si la distribution des valeurs diffère d'un patient à l'autre (faire des histogrammes par exemple).

5. une fois les données prêtes, les répartir en 3 sets en les stockant séparément via np.zeros (ou np.memmap si les données sont de taille importante). Souvent, on aura *input_train*, *output_train*, *input_val*, *output_val*, *input_test* et *output_test*.

6. téléchargez un U-net qui marche avec tensorflow (et non pytorch ni caffe).

   `https://github.com/zhixuhao/unet/blob/master/model.py`

7. 3 réseaux (de type U-Net) seront à calibrer:

   - Pour l'apprentissage de notre réseau 1: on voudra savoir segmenter la zone $10 \cup 150 \cup 250$, donc ces valeurs seront à mettre à 1, et les zéros resteront tels quels, tout cela sachant les T1 et T2, donc 2 inputs.
   - Pour l'apprentissage de notre réseau 2: on voudra savoir segmenter la zone $150 \cup 250$, sachant $10 \cup 150 \cup 250$ versus 0, ainsi que les T1 et T2, donc 3 inputs.
   - Pour l'apprentissage de notre réseau 3: on voudra savoir segmenter la zone 250, sachant $150 \cup 250$ versus le reste et $10 \cup 150 \cup 250$ versus le reste (2 inputs en plus des T1 et T2).

8. Commençons par l'étape 1:

   - paramétrez ce modèle pour qu'il puisse avoir le nombre d'entrées voulues, et une sortie entre 0 et 1 (normalement la sortie est deja réglée pour sortir uen probabilité pixelwise),
   - entrainez votre réseau comme vu en cours (model.fit) avec un earlystopping (pour qu'il s'arrete quand il a assez convergé), un checkpoint (pour sauvegarder les meilleures version de votre réseau). Note: vous pouvez utiliser un $fit\_generator$ si vous voulez faire de l'augmentation de données mais a priori ce n'est pas utile ici.
   - faites les predictions sur les données tests (on seuille les outputs à 0.5 pour qu'elles vaillent soit 0 soit 1) et calculez le dice moyen correspondant avec les verités terrains. Attention, la moyenne se calcule normalement par patients, et non pas par coupes (au cas ou vous auriez utilisé l'approche 2D). Rappel du dice:

     $$Dice(X, Y) = \frac{2 * |X \cap Y|}{|X| + |Y| + \epsilon}$$

     avec $\epsilon = 10^{-6}$ par exemple (on veut éviter la division par zéro).
   - n'oubliez pas de sauvegarder votre réseau (poids *et* modèle) pour une utilisation ultérieure,

9. maintenant que vous savez différencier $10 \cup 150 \cup 250$ des zéros, vous pouvez passer a l'apprentissage du deuxième réseau, puis du troisième.

10. une fois vos trois réseaux combinés, vous pouvez enfin faire les prédictions finales (un tout petit calcul mathématique sera nécessaire pour savoir prédire les labels finaux (exemple: si le réseau 1 prédit 1, et le deuxième prédit 1, mais le troisième prédit 0, c'est que l'on a la zone labélisée 150, etc).

2 Compétition BraTS

Maintenant que vous savez résoudre des challenges de segmentation, à votre tour de faire le challenge de segmentation de tumeurs cérébrales ! Attention à bien gérer la quantité de données (y aller progressivement, ne chargez pas tout d'un coup, et pensez à memmap !).

Pour l'énoncé:
`https://www.med.upenn.edu/cbica/brats2020/data.html`
Pour les données, téléchargez les dossiers "HGG" et "LGG" dans le dossier dataset indiqué dans le README à l'adresse suivante: `https://github.com/as791/Multimodal-Brain-Tumor-Segmentation/blob/master/README.md`

3 Ressources supplémentaires

`https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks#`