0.1 Mode d'emploi de UNETR

Ce mode d'emploi nécessite l'installation de notre code de UNETR sur votre ordinateur

0.1.1 Installation des librairies

Le code UNETR est fourni avec un fichier requirements.txt qui compile toutes les librairies, et leurs versions, nécessaires à l'utilisation du code. Ce fichier est situé à la racine du code. Pour installer ces librairies, il vous faut exécuter le code suivant à la racine. Cette ligne de code est aussi présente dans le fichier "LancerIAOrdi.ipynb".

```
pip install -r requirements.txt -f
↪  https ://download.pytorch.org/whl/torch_stable.html
```

0.1.2 Transformation des données au format Dicom vers Nifti

Avant de lancer l'entraînement, les données doivent être converties en Nifti. Pour ce faire, il vous faudra exécuter le code présent dans le fichier "converteur.ipynb". Ce code nécessite d'autres versions des librairies que celle de UNETR, il faudra donc utiliser un environnement différent et exécuter la ligne ci-dessous.

```
pip install -r requirementsConverteur.txt -f
↪  https ://download.pytorch.org/whl/torch_stable.html
```

Deux changements peuvent être apportés à ce fichier. Dans la troisième cellule, vous pouvez changer le nom du dossier d'où proviennent les images Dicom (meta_data_dir) et dans la septième le nom du dossier où seront sauvegardées les images Nifti (saving_dir).

0.1.3 Entraînement

Une fois les librairies installées et la conversion des données faite, vous pouvez exécuter la ligne de commande ci-dessous.

```
os.environ["WANDB_CACHE_DIR"]="ArtifactsRunAll5"
os.environ["WANDB_DISABLED"]="true"
!python unetr/cli.py -c config.yaml fit --trainer.logger.name="nomDuTest"
↪  --data.data_dir="EmplacementDataNifti" --data.train_batch_size=16
↪  --data.workers=8 --data.val_batch_size=16
```

Plusieurs modifications peuvent être réalisées, vous pouvez changer le nom du fichier où se sauvegarderont toutes les données de l'entraînement ici nommé "nomDuTest" veillez à ce que ce nom soit différent de ceux déjà présents à la source du programme, le cas contraire amènera une erreur. Vous pouvez modifier le nom du dossier où se trouvent les images Nifti, ici "EmplacementDataNifti", il est le même que celui donné à la variable saving_dir du "converteur.ipynd".

Je vous conseil de ne pas modifier les variables data.train_batch_size, data.workers et data.val_batch_size, les augmenter provoquerait une surutilisation du GPU de la machine et donc un arrêt prématuré de l'entraînement.

Dans le fichier "config.yaml", voici la liste des variables que vous pouvez modifier pour améliorer les résultats du modèle :

- max_epochs : nombre d'époques maximal réalisé par l'entraînement
- min_epochs : nombre d'époques minimal réalisé par l'entraînement
- lr : le learning rate du modèle
- random_flip_prob : probabilité de faire une mise en miroir des images sur l'axe des y
- unfreeze_backbone_at_epoch : jusqu'à cette époque, le modèle entraîne la dernière couche sur nos données en fonction des points du modèle pré-entraîné.

0.1.4 Application

Pour appliquer le modèle à des données, vous pouvez utiliser le fichier "ApplicationUNETR.ipynd" en exécutant la septième cellule (ainsi que les précédentes). Il faudra ajouter un point dans le fichier "model_module.py" devant la ligne 26 et 27 (voir ci-dessous).

```
#pour l'entrainement
from dice_bce_loss import DiceBCELoss
from networks.unetr import UNETR
#pour l'application
from .dice_bce_loss import DiceBCELoss
from .networks.unetr import UNETR
```

Ce fichier vous permet d'appliquer le modèle UNETR présent à l'adresse indiquée à la variable "pathModelFile", sur un ensemble d'images en Dicom présentes dans le dossier indiqué dans la variable "meta_data_dir" et de calculer le dice, la spécificité et la sensibilité associés aux labels. Il est possible d'afficher ou de sauvegarder les labels prédits en récupérant la variable "labelPred" calculée aux lignes 48 et 57. Il vous est aussi possible d'utiliser le logiciel MetIA.

0.1.5 Synchronisation avec Wandb

Une fois l'entraînement fait, il est possible de visualiser les courbes et les résultats sur Weights and Biases. Pour ce faire, copier les dossier de Run et d'Artefact créé pendant l'entraînement au même endroit sur votre ordinateur. Placez vous à l'endroit ou se trouve les dossiers et ouvrez un terminal puis, récupérer le chemin du fichier wandb contenu dans Run/logs/wandb/offline-run[...]/run-[...].wandb. Puis appliqué la commande suivante dans le terminal.

```
import wandb
wandb.login()
wandb sync Run/logs/wandb/offline-run[...]/run-[...].wandb
```

0.1.6 Visualiser les résultats

- aller sur le wandb où a été synchronisé le modèle
- cliquer sur Projects

— sélectionner le projet à visualiser (vous arriverez sur une page avec tout les run sur votre gauche et des courbes à droites, en fermant ou en ouvrant les yeux à gauche des noms de run vous pourrez afficher ou non leurs courbes)
— pour voir les résultats en images, aller sur Artifacts dans la colonne toute à gauche de la page
— Trouver RUN_TABLE en gris sur la colonne où se trouvais le nom des run
— Cliquer sur la run table que vous voulez et la version que vous voulez, aller dans Files en haut de votre page et en dessous du nom de votre run table
— cliquer sur val_table.table.json
— en cliquand sur les images vous pourrez les agrandir et utiliser les boutons other et meta pour dés-afficher les labels.

0.1.7 À savoir

Si vous utilisez les fichiers préprogrammés, "converteur.ipynb" et "ApplicationUNETR.ipynb", vos données doivent être rangées comme suit : un dossier contenant un dossier pour chaque patient. Chaque dossier de patient doit contenir des fichiers débutants par MR rangés ou non dans un dossier RM et un fichier débutant par RS rangé ou non dans un dossier META.

0.2 Mode d'emploi de MetIA

Ce mode d'emploi nécessite l'installation de notre code de MetIA sur votre ordinateur

0.2.1 Installation des librairies

Le code MetIA est fourni avec un fichier requirements.txt qui compile toutes les librairies, et leurs versions, nécessaires à l'utilisation du code. Ce fichier est situé à la racine du code. Pour installer ces librairies, il vous faut exécuter le code suivant à la racine du code, dans un terminal.

```
pip install -r requirements.txt -f
↪  https ://download.pytorch.org/whl/torch_stable.html
```

0.2.2 Exécution

Ouvrir un terminal à la source du code (là ou se trouve le fichier MetIA.py) et taper la commande suivante :

```
python MetIA.py
```

0.2.3 Utilisation

Pour changer le modèle, allez dans logiciel puis, TrainedModel et remplacer le fichier présent par votre modèle.

Lorsque vous êtes sur l'application, appuyer sur le bouton en haut à gauche pour ouvrir le dossier du patient que l'on souhaite analyser. Ce dossier contient un dossier IRM avec toutes les slices de l'IRM et un dossier RTSTRUCT ou, des fichiers dicoms commençant par MR pour les IRM et RS pour le RTSTRUCT.

Lorsque les images se sont affiché au centre de l'application, vous pouvez sauvegarder un fichier Excel qui vous indiquera l'emplacement, la taille et la couleur sur les slices des métastases et un fichier RTSTRUCT qui, si il est créé dans un dossier contenant un RTSTRUCT commançant par RS, récupérera les données de ce dernier et créera un fichier "RS$_u pdated.dcm$"$sinonunfichier$"$RS_MetIA.dcm$".

Chargement sur Raystation

- charger le patient et ses IRM sur Raystation
- créer un nouvel RTSTRUCT avec les zones d'intérêt hors tumeurs
- exporter le patient et ses IRM dans un dossier
- mettre les IRM dans un dossier IRM et le rtstruct dans un dossier à part
- lancer MetIA sur le dossier patient (ou se trouve le dossier IRM et le dossier avec le rtstruct)
- sur MetIA enregistrer le rtstruct dans le dossier avec le rtstruct du patient
- dans Raystation supprimer le Rtstruct précédement créé et ajouter le rtstruct rt_update.dcm qui se trouve dans le dossier avec le rtstruct
- retraivailler les contourages (une IA ne peut être considérée comme aussi fiable qu'un expert)