
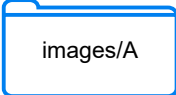
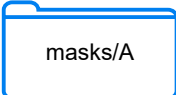
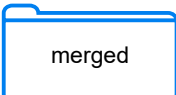

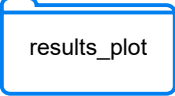

















Project Architecture

 data	Contains all the data used to train the model
 images/A	Contains the OCT-scans used to train the model
 masks/A	Contains the segmented masks used to train the model
 merged	Contains the 3 layer (ILM, BM, CSI) merged but not segmented yet
 logs	
 results_plot	Contains the plot images of the results images after a prediction
 weights	Each time a model is being trained, the weights will be saved here
 install.sh	All the dependencies needed for this project are listed in requirements.txt and can be automatically installed using install.sh
 requirements.txt	
 README.md	JOURNAL.md contains informations about multiple iterations of training of the models
 JOURNAL.md	
 Init.ipynb	Can be used to run the training, prediction and visualize the image predicted
 Box_plot_history.ipynb	Plots the loss and performance metrics
 Image_augmentation_verifier.ipynb	Used to output augmented images to find the best parameters
 Learning_Rate_Finder.ipynb	Used to find the best learning rate parameter
 config.py	Saves the configuration for the data, hyperparameters, model and history of the model
 helper.py	Contains various methods provide functionality (like loading the data)
 segmentation.py	This script can be used to generate new segmentation masks in case of new data provided
 model.py	The U-net model architecture is defined here along with the metrics
 train.py	This script can be used to train the model A and model B and perform a prediction at the end of training
 predict.py	Here is defined the method to do prediction, it can also be run to make a new prediction