REUNION CLIPS
27/02

PRESENTATION DU TRAVAIL

1. Réussite du fonctionnement du programme sans Docker
   → Assez long : si 2 fichiers PET, le programme essaie de segmenter l'un puis l'autre
   → Réussi à utiliser Slicer pour visualiser les segmentations
   → Prochaine étape : Produire les segmentations manquantes et obtenir les métriques manquantes
   → 1er test effectué sur 11011101221002 (vérifier le résultat mais semble ne pas fonctionner)

2. Lecture du code d'entrainement : problème de documentation

PRESENTATION DU TRAVAIL

```python
class TrainerV2:
    def __init__(
            self,
            callbacks_per_batch=None,
            callbacks_per_batch_loss_terms=None,
            callbacks_per_epoch=default_per_epoch_callbacks(),
            callbacks_pre_training=default_pre_training_callbacks(),
            callbacks_post_training=default_post_training_callbacks(),
            trainer_callbacks_per_batch=trainer_callbacks_per_batch,
            run_epoch_fn=epoch_train_eval,
            logging_level=logging.DEBUG,
            skip_eval_epoch_0=True):
        """

        Args:
            callbacks_per_batch:
            callbacks_per_batch_loss_terms:
            callbacks_per_epoch:
            callbacks_pre_training:
            callbacks_post_training:
            trainer_callbacks_per_batch:
            run_epoch_fn:
            skip_eval_epoch_0: if ``True``, validation/test will not be run for epoch 0
        """
```

```python
class TrainerV2:
    def fit(self,
            options,
            datasets,
            model: nn.Module,
            optimizers_fn,
            losses_fn=default_sum_all_losses,
            loss_creator=create_losses_fn,
            log_path=None,
            with_final_evaluation=True,
            history=None,
            erase_logging_folder=True,
            eval_every_X_epoch=1) -> RunMetadata:
        """
        Fit the model

        Args:
            options:
            datasets:  a functor returning a dictionary of datasets. Alternatively, datasets infos can be specified.
                       `inputs_fn` must return one of:

                       * datasets: dictionary of dataset
                       * (datasets, datasets_infos): dictionary of dataset and additional infos

                       We define:

                       * datasets: a dictionary of dataset. a dataset is a dictionary of splits.
                         a split is a dictionary of batched features.
                       * Datasets infos are additional infos useful for the debugging of the
                         dataset (e.g., class mappings, sample UIDs). Datasets infos are
                         typically much smaller than datasets should be loaded in
                         loadable in memory
            model: a `Module` or a `ModuleDict`
            optimizers_fn:
            losses_fn:
            loss_creator:
            log_path: the path of the logs to be exported during the training of the model.
                if the `log_path` is not an absolute path, the options.workflow_options.logging_directory
                is used as root
            with_final_evaluation:
            history:
            erase_logging_folder: if `True`, the logging will be erased when fitting starts
            eval_every_X_epoch: evaluate the model every `X` epochs
```

A-t-on des nouvelles de Vancouver ?

PRESENTATION DU TRAVAIL

```
325
326         # instantiate the optimizer and scheduler
327         logger.info('creating optimizers...')
328         if optimizers_fn is not None:
329             optimizers, schedulers, per_step_scheduler_fn = optimizers_fn(datasets, model)
330             logger.info('optimizers created successfully!')
331         else:
332             logger.info('optimizer fn is None! No optimizer created.')
333             optimizers, schedulers, per_step_scheduler_fn = None, None, None
334
```

DATE DE PROCHAINE RÉUNION ?