• Summary

In my solution, I use well-known U-Net (https://arxiv.org/abs/1505.04597) like encoder-decoder Neural Network architecture for semantic segmentation.

I have used ensemble of four models trained with three input resolutions:

- 768 * 768
- 1024 * 1024
- 1472 * 1472.

Three pretrained CNN encoders (from timm: https://github.com/rwightman/pytorch-image-models):

- efficientnet_b7
- convnext_large
- tf_efficientnetv2_l

And one transformer encoder:

- coat_lite_medium

Main contributions:

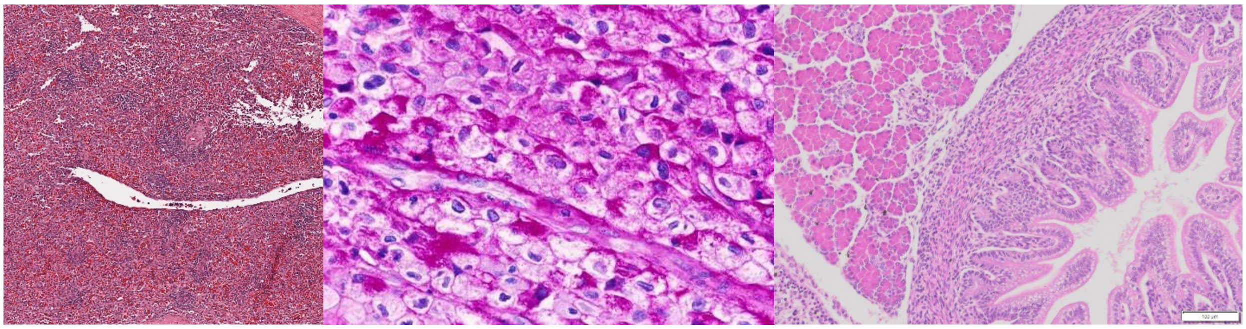
- External data
- Pseudo labeling
- Heavy augmentation
- Color transferring on training data
- Data sampling
- Predicting pixel_size and organ for generalization

• Data Preparation

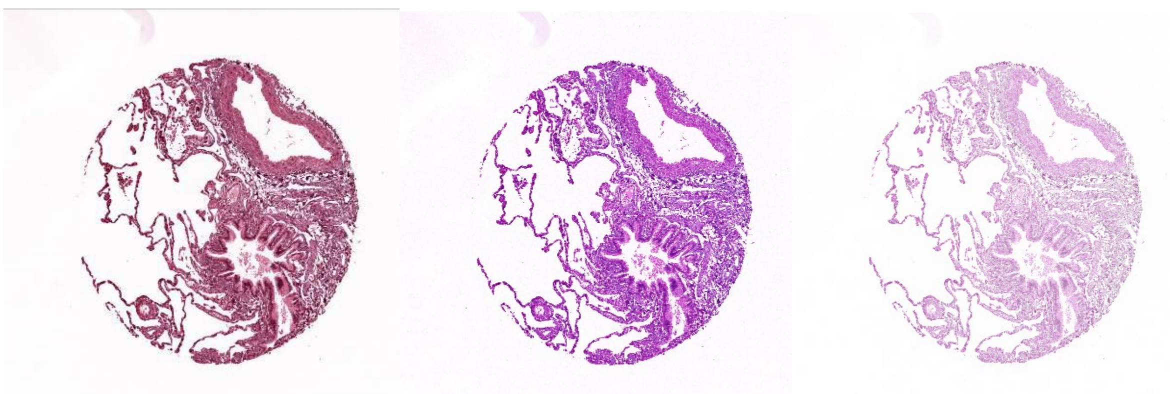
External data helped models to be ready for new unseen datasets. I have downloaded some HPA data (with this notebook https://www.kaggle.com/code/carnozhao/hpa-data-download) and picked some images manually from sources posted here and previous Hubmap and Panda competitions.

Training images recolored with Staintools (using this great notebook https://www.kaggle.com/code/gray98/stain-normalization-color-transfer) with 3 different target images and used with 15% chance instead of original during training.

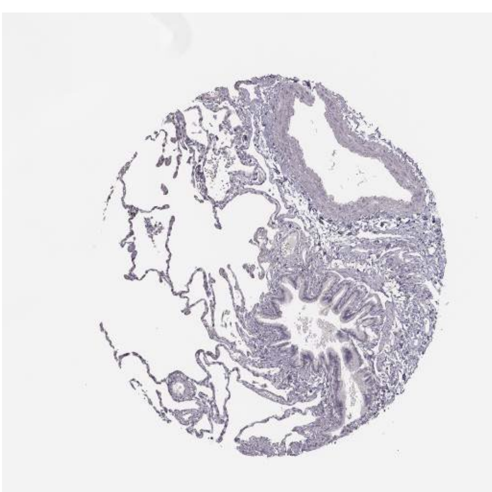
As targets used available test image, and found examples of PAS and H&E stains:



Example of transferred image from training data:



Original image:



(Probably a good idea to use this tool as additional Test-Time Augmentation (TTA). However, I have not tried it during the contest.)

All external data pseudo-labeled using ensemble of initial models.

In addition, training data was pseudo-labeled and used as ground truth for training with 30% chance.

- ## Training Method(s)

Training data separated to 5 folds. Stratified by sex/age and organ evenly.

Models with different encoders trained on three input resolutions: 768 * 768, 1024 * 1024, 1472 * 1472 with 5 folds. Largest input resolution give best score.

Models also trained to predict organ and pixel_size. I think these aux outputs help to train more robust model. pixel_size calculated for resized input resolution and changed during training augmentations.

Heavy augmentations used:

- random cropping/padding
- scaling
- rotating
- flipping
- color changing
- blur/noise
- saturation/brightness/contrast
- elastic

Training parameters:

- Loss: combined loss function which consists of Dice and Focal loss (https://arxiv.org/abs/1708.02002) for segmentation, BCE for organ classification, MSE for pixel_size: dice_focal + 0.1 * bce + 0.1 * mse
- Optimizer: Adan (https://arxiv.org/abs/2208.06677)
- lr: 0.0001 with ReduceLROnPlateau scheduler
- half-precision (float16) used to reduce GPU memory
- batch size: 3-8 (depends on encoder and resolution)

Test-time augmentations used on validation (flip, crop, padding) + external data also separated on folds and used as validation to get best checkpoints.

4 TTA used on test prediction: flip * 2 rotations to 90 degrees.

Thresholds tweaked using public LB. For lung it is very low, only 0.06.

- ## Simple Features and Methods

Coat performed the best as single model, but ensemble with CNNs scored more. 768 or 1024 resolution enough. So, if fast inference required, better to train a single Coat model on lower resolution and predict without TTA.

- ## Model Execution

Hardware:

System with 2 x NVIDIA RTX A6000 (48 GB). Large GPU memory required for high-resolution models.

Software:

Ubuntu 22.04 LTS
Python 3.9 (Anaconda installation)
CUDA 11.6
torch and other libs from requirements.txt

Training time:

~ 1 week for all models total using one a system with 2 x NVIDIA RTX A6000 (48 GB)

Test prediction time:

~7-8 hours for final submission

- **References**
  - Timm models: https://github.com/rwightman/pytorch-image-models
  - Adan: https://arxiv.org/abs/2208.06677
  - U-Net: https://arxiv.org/abs/1505.04597
  - Focal loss: https://arxiv.org/abs/1708.02002
  - Color transferring notebook: https://www.kaggle.com/code/gray98/stain-normalization-color-transfer
  - Download HPA data: https://www.kaggle.com/code/carnozhao/hpa-data-download
  - For external data: https://pathdb.cancerimagingarchive.net/imagesearch
  - Prostate cANCer graDe Assessment (PANDA) Challenge: https://www.kaggle.com/competitions/prostate-cancer-grade-assessment
  - HuBMAP - Hacking the Kidney: https://www.kaggle.com/competitions/hubmap-kidney-segmentation