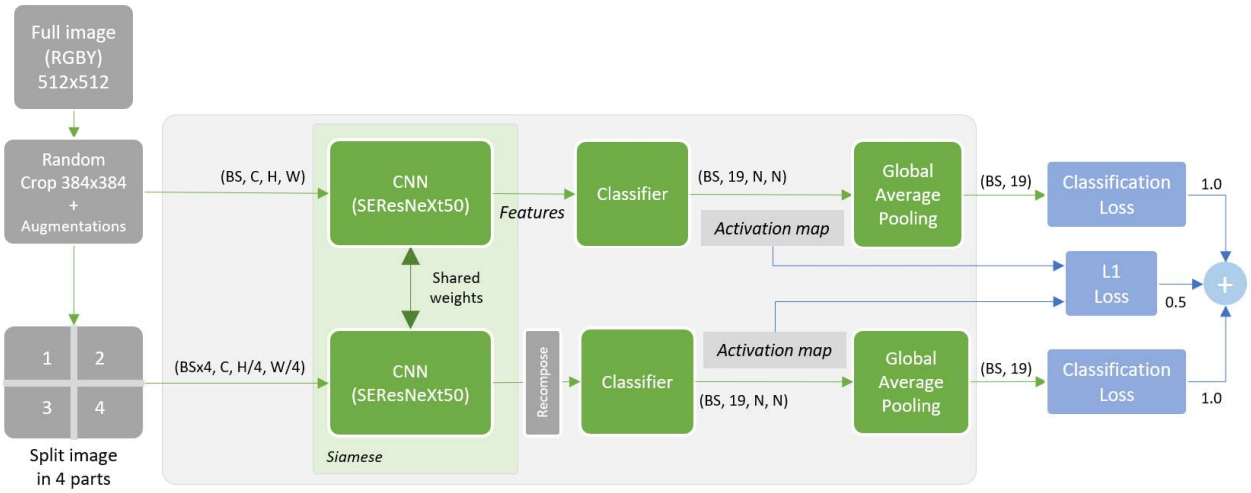
Introduction

Early in this competition I decided to start with WSSS (Weakly-Supervised Semantic Segmentation) SOA approaches to learn something new. Full image training with multi labels and try to build a segmentation. Many solutions are based on CAM utilization. In short, at the end of the pipeline, make CAM grow and spread over the image on the regions of interest. However, as cell masks were already available, I didn't need to go with the full WSSS process and just intersect at some point CAM with cells to get predicted labels. I've experimented with both Puzzle-CAM [1] and DRS (Discriminative Region Suppression) [2] approaches. I finally got best results with a modified Puzzle-CAM trained on RGBY images and with a two stages inference (full image and per cell basic ensemble).
I teamed up early with ZFturbo that used a different approach and both solutions ensembled quite well. Next step to improve was to generate OOF (per cell prediction for each class) from some of my best model variants that could benefit ZFTurbo's models and then to our ensemble.
Close to the end of competition we had room left on inference (our pipeline was running in around 6h30 only) and we agreed that integrating new models should help and we merged with Dieter that brought new models/ideas that made the difference to be in the money zone.

Training

Siamese network with CNN backbone followed by classifier to build activation maps per class. GAP moved that end to get full image predictions. Combo loss that takes into account both full and puzzled/recomposed features and distance between activation maps.

Pipeline:

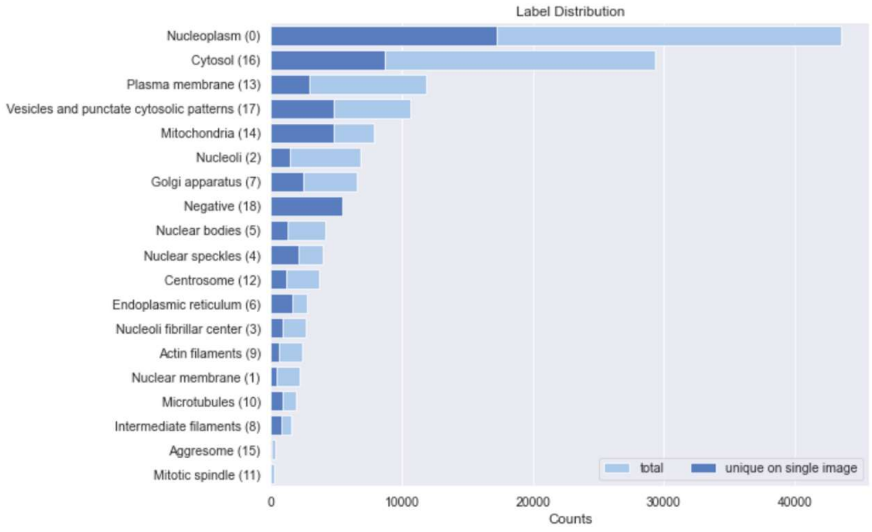


Notice the Classifier/GAP swap used in WSSS compared to regular classifiers and L1 loss for overall consistency.

Augmentations:
Simple augmentations that apply to RGBY images:

- Noise: GaussNoise, CoarseDropout, IAAAdditiveGaussianNoise
- Flips/Rotations: HorizontalFlip, RandomRotate90
- Rotate/Distorsion: GridDistortion, ShiftScaleRotate, ElasticTransform, OpticalDistortion, IAAAffine/Shear
- Blurs: GaussianBlur, MotionBlur, MedianBlur
- Colors: RandomGamma, RandomBrightnessContrast

Sampling strategy:
Classes distribution is depicted below. Classes are quite imbalanced especially for Aggresome (class 15) and Mitotic spindle (class 11). Nucleoplasm (class 0) and Cytosol (class 16) are over-represented.



A weighted random sampling has been applied to balance each class during training. Basically, less weight for nucleoplasm and more weights for Mitotic spindle. However, to counterbalance the rare classes 11 and 15 important oversampling due to this strategy, we've clipped weights value to the one computed on Intermediate filaments (class 8).

Data:
Train set and external data had been used to train different model variants:

- 89k images: Train set + External public data shared by host
- 98k images: Train set + External public data shared by host + HPA 2018

Each single layer image has been merged and resized to create RGBY 512x512 images (protein + context). A few similar images have been found on sanity check but not removed.
External data was key for this competition to increase volume and make models better.

Cross validation:
Train dataset is split in 4 folds with a MultilabelStratifiedKFold strategy.

Fold	1	2	3	4	total
Nucleoplasm	9706	9705	9705	9705	38821
Nuclear membrane	517	517	517	517	2068
Nucleoli	1570	1570	1570	1571	6281
Nucleoli fibrillar center	614	615	614	614	2457
Nuclear speckles	908	908	908	908	3632
Nuclear bodies	955	956	955	956	3822
Endoplasmic reticulum	599	600	599	599	2397
Golgi apparatus	1440	1439	1439	1440	5758
Intermediate filaments	366	366	367	367	1466
Actin filaments	566	566	566	566	2264
Microtubules	459	460	460	460	1839
Mitotic spindle	53	52	52	52	209
Centrosome	862	862	862	862	3448
Plasma membrane	2605	2605	2604	2604	10418
Mitochondria	1794	1795	1794	1794	7177
Aggresome	87	88	88	87	350
Cytosol	6828	6827	6827	6827	27309
Vesicles and punctate cytosolic patterns	2319	2319	2319	2318	9275
Negative	1222	1223	1222	1223	4890

Training procedure:
4 main models trained with different backbones (but only variants of #1 in the final ensemble).

1. seresnext50_32x4d
2. gluon_seresnext101_32x4d,
3. cspresnext50
4. regnety_64

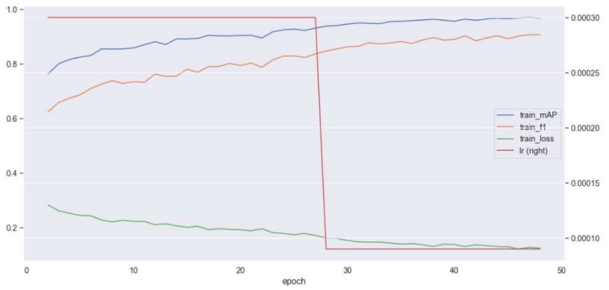
Weighted ComboLoss:

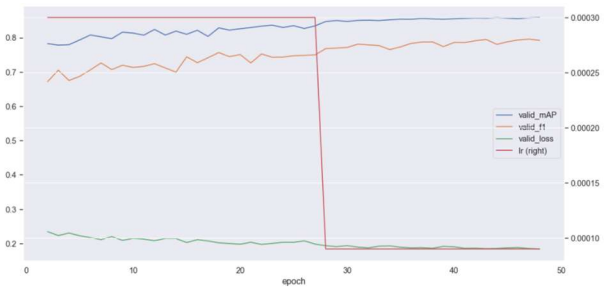
- BCEWithLogitsLoss (each with weight=1.0)
- L1Loss (weight from 0.25 to 0.5 got good results)

LR and hyperparameters:

- Optimizer: Adam, LR= 0.0003, beta1=0.9
- LR scheduler: ReduceLROnPlateau, factor = 0.3, patience = 8
- Epochs: 48
- Batch size: From 32 to 36
- FP16 enabled
- Single cycle

Criteria to save model's weights is based on best ComboLoss only, F1 and mAP scores are just monitored:

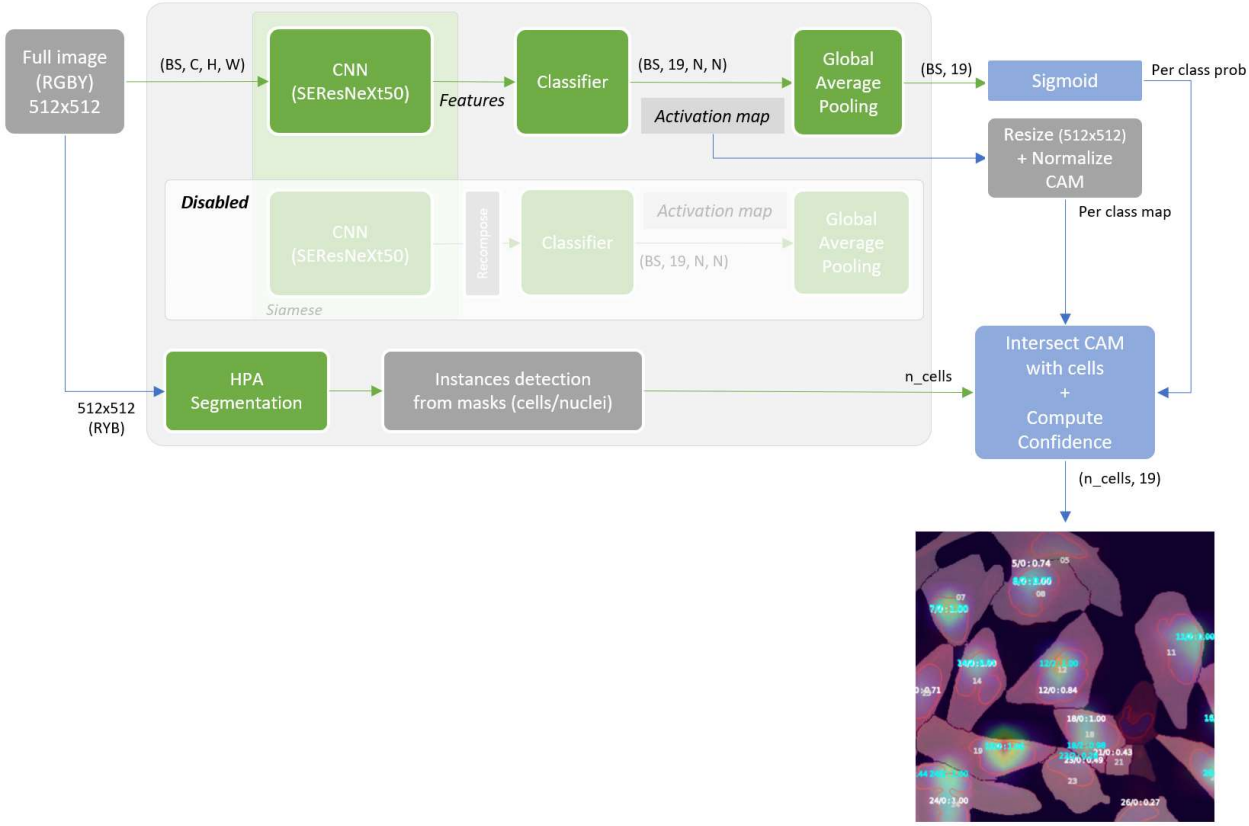




Inference

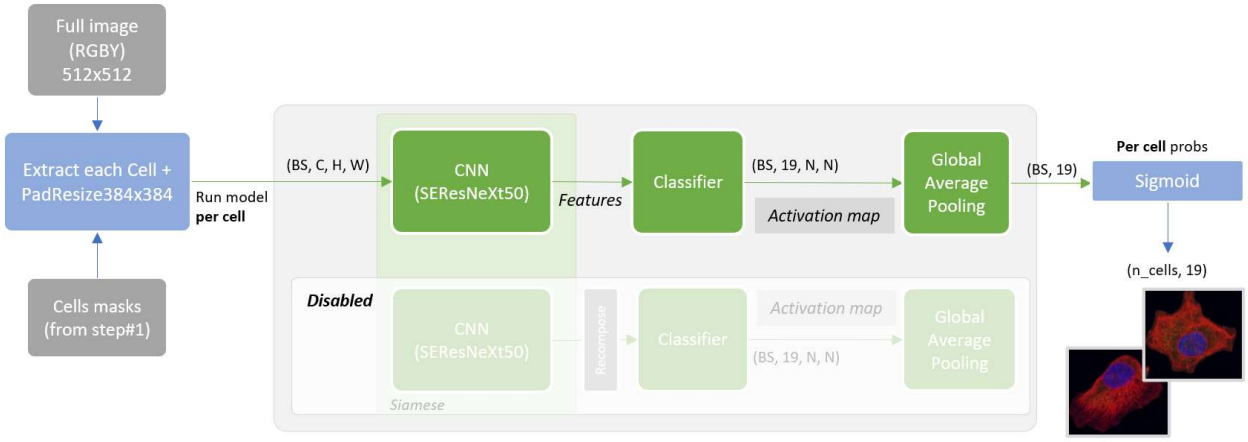
Inference is based on two steps ensembled at the end.

Step #1:
It's the inference related to a model trained with activation maps output normalized to [0-1.0] range and resized to input image size. HPA segmentation pipeline is executed in parallel to get cell masks instances. Each cell is intersected with CAM (overlap + magnitude score) and weighed with the per-class probability outputted from the sigmoid.



Note: HPA segmentation pipeline provided by the host has been modified to make it faster (around x6). As the host recommended scale factor = 0.25, it globally matches with 512x512 images and we've modified the morphology post-processing to detect instances based on 512x512 and not on original image size. The gain is on speed but we lose on quality, some cells remain merged with this optimization but they should be split. The impact on score has been estimated to around -0.002 which is an acceptable trade off.

Step #2:
It is another inference that comes for free by simply re-using the model and fed by each cell instance (from stage#1) crop.



Stage#1 vs stage#2 predictions are different by design, stage#1 are sparse whereas stage#2 are more flatten. Both acting together gave some regularization to the results.

From a performance point of view, this single model (both stages + HPA segmentation) on 559 images ran in around 2h with one P100 GPU. Additional averaging with different model variants (different input dataset, different backbones, seed) got public LB around 0.52/0.53.

Misc

What could be improved?
DRS approach was also giving interesting results, the single model (with vgg16 backbone) was slow to train and got a lower score compared to Puzzle-CAM (but acceptable). Inference was slow too that's why we didn't get a chance to ensemble it at the end but it might be worth assessing as CAM generated are different.

What did not work (or not better than this solution)?
- YoloV5 model based on OOF and RGB image
- Post process probabilities to move to rank probabilities on ensemble
- TTA (it worked indeed but gave tiny improvement)
- Larger full image

Bibliography

[2]: Beomyoung Kim, Sangeun Han, Junmo Kim. “Discriminative Region Suppression for Weakly-Supervised Semantic Segmentation.” 2021. https://arxiv.org/pdf/2103.07246.pdf.

[1] Jo, Sanhyun and Yu, In-Jae. “Puzzle-CAM: Improved localization via matching partial and full features.” *arXiv preprint arXiv:2101.11253*, 2021. https://arxiv.org/pdf/2101.11253.pdf.