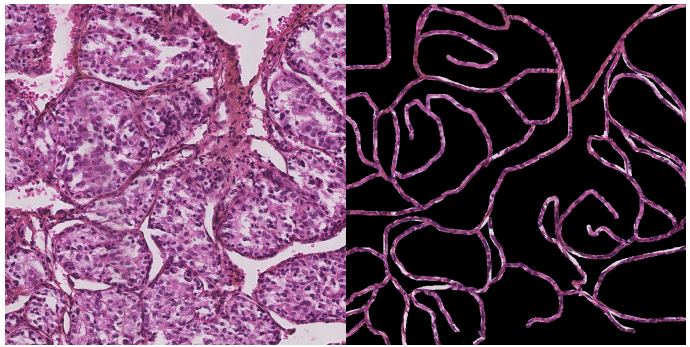
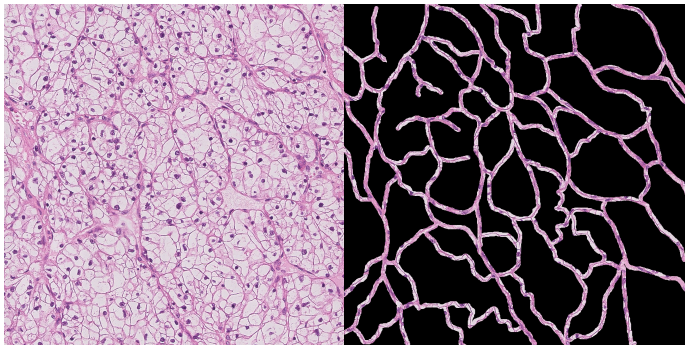
CNN-GCN comparison on renal cancer dataset

The problem context

Given a dataset of samples (image, segmentation_mask)

Compare CNN and GCN neural network performance on the supervised classification of **papillary** and **clear cell** renal cancer:
-CNN are trained on **raw data**
-GCN are trained on **UNET segmentation masks**





Definition of the framework of the comparison

- Classify WSI slide patches that represent one of two categories: [**clear cell** or **papillary**] renal cell cancer.
- In the context of convolutional neural networks, the classification is done on raw images.
- On the other hand, classification on graphs has to be properly defined, since no graph data structure is readily available in the dataset.

- To achieve this, **the classification performance in the case of graph convolutional networks is evaluated on graph samples that are defined as:**

  The **graph (node, edges) data structure extractable from the vascular network** of a given wsi slide patch:
  The **edges are the blood vessels**, and the **points of connection between them are the nodes**.
  In this context, self loops are possible

- Thus, the selected approach for classification on graph convolutional networks is sample graph classification(note that other approaches are possible, such as node classification, with an appropriate alternative definition of the samples)

The dataset

Composed of:
- Train split
- Test split

Each split contains:
- **Paired (image, mask) samples**, where the mask is the segmentation of the **vascular network**

In the training set, **patients and their respective annotations are present (useful for patient-wise image standardization).**
The test set provides **only the paired samples.**

Note, some images in the train set do not have a matching mask and some masks do not have the matching image: all these have been removed

Raw data, Segmentation masks and Graph data

- Raw data is represented by image patches coming from the **Whole Slide Images** of a given annotation group of a patient.

- Segmentation masks highlight the vascular network of the corresponding image.

-Graph data has been extracted from the segmentation masks, in such a way that it represents the segmented vascular network:

- **nodes** are the points of intersection between different vessels

- **edges** are the vessels shown in the segmentation mask

How graph samples are created

In order to train Graph Convolutional Networks, graph samples are required.
In this work, this is achieved by extracting, for each (image, mask) pair, the graph structure of the segmentation mask.

The procedure is the one proposed in [1], with adaptations (described in the code documentation of the ToGraphTransform class in lib.data.processing_utils.py module) **to reduce its time complexity.**

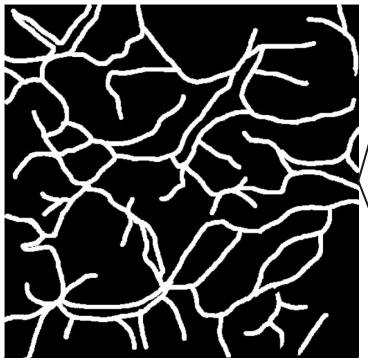
Phase 1
At an high level, the procedure uses a **specific sequence of image morphological operations to extract the image representation of all nodes, edges and joints** (intersection between a given edge and and node). The result are three images each of which contain only nodes, edges and joint respectively.

Phase 2
After this first phase, each group of pixels representing a node, edge and joint is labeled separately by means of **connected component clustering.**
This allows to **abstract from the pixel representation** and to find a way to represent node, edge and joint objects.
Furthermore, each **connected component provides descriptive geometric features relative to that object, such as its bounding box and its area. These will be used as node features for the classification task.**

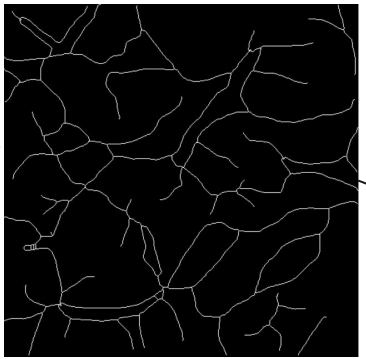
[1] Automatic Extraction of Graph-Like Structures from Binary Images
https://www.researchgate.net/publication/220872217_Automatic_Extraction_of_Graph-Like_Structures_from_Binary_Images

How graph samples are created

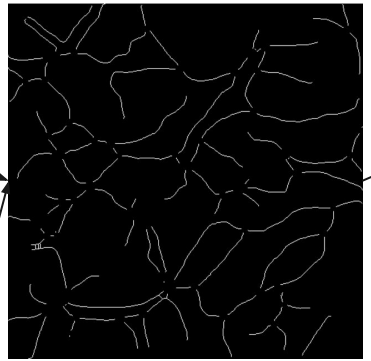
Original mask



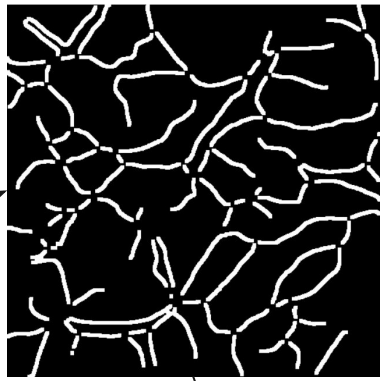
Skeletonized mask



Skeletonized without nodes



Skeletonized w.o. Nodes and expanded (dilation)

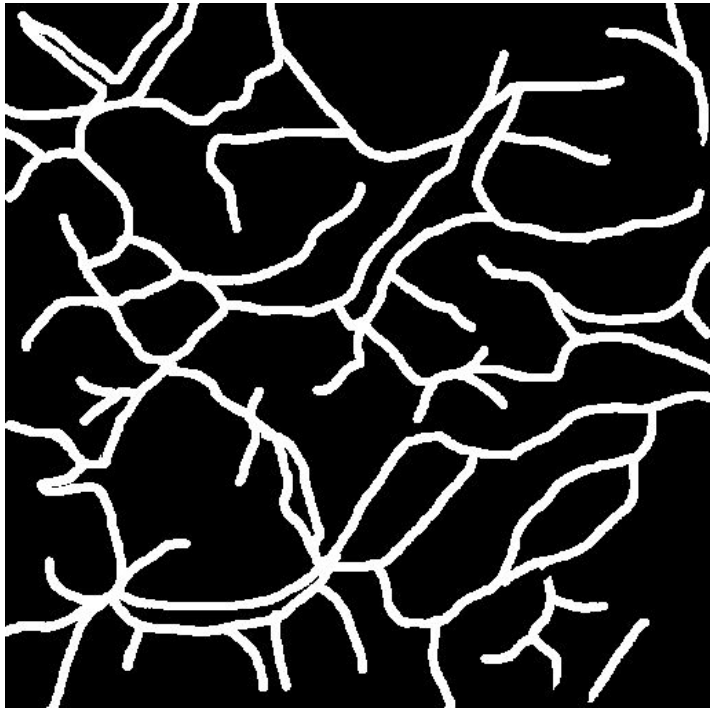


Nodes

Joints (intersection of edges and nodes)

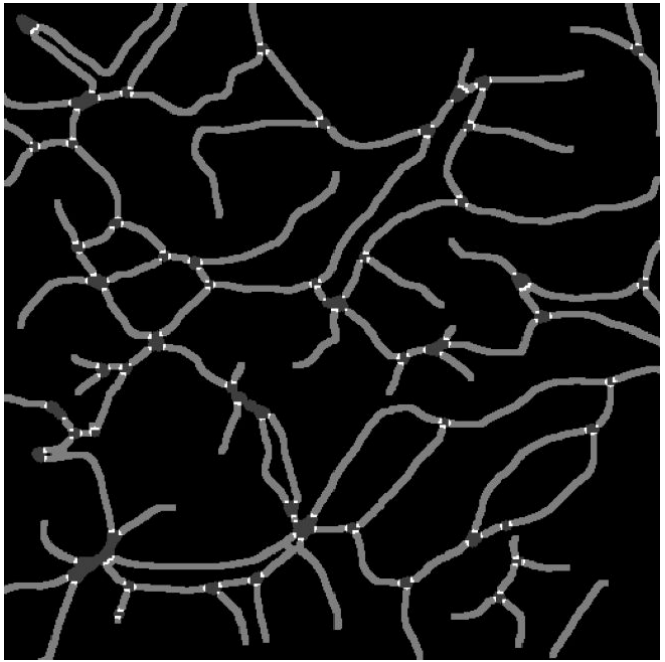
[1] Automatic Extraction of Graph-Like Structures from Binary Images
https://www.researchgate.net/publication/220872217_Automatic_Extraction_of_Graph-Like_Structures_from_Binary_Images

How graph samples are created

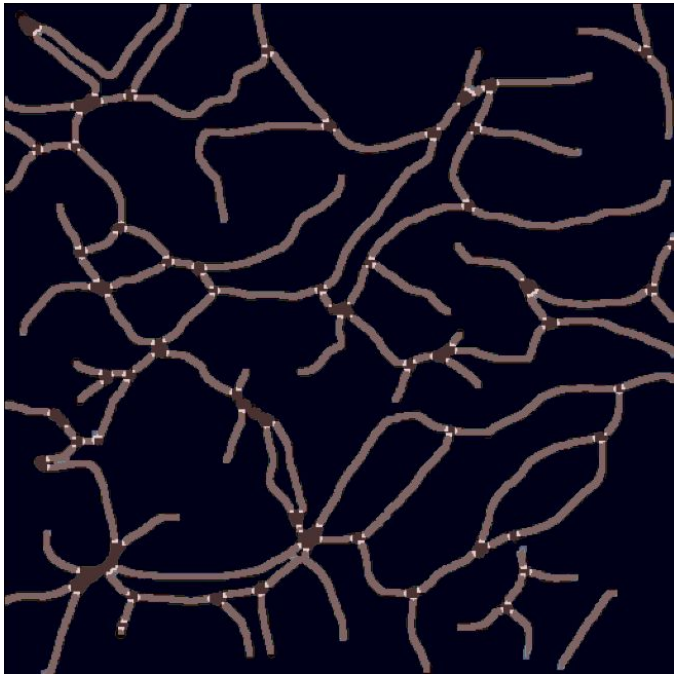


How graph samples are created

Nodes(dark gray), Edges (gray), Joints (white)

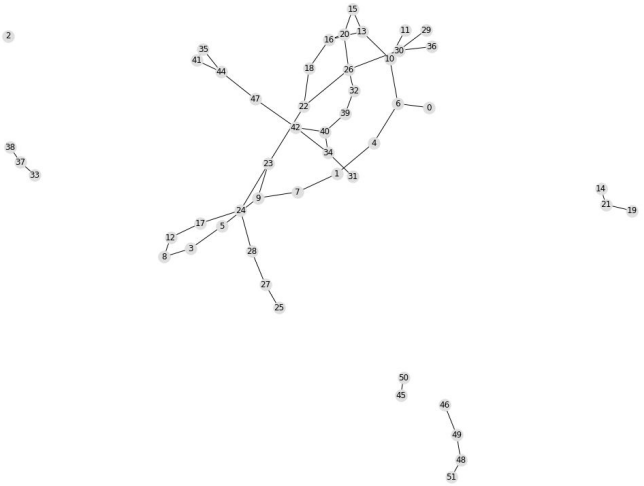


Overlap of the found components and the true mask(slight red color is the true mask)

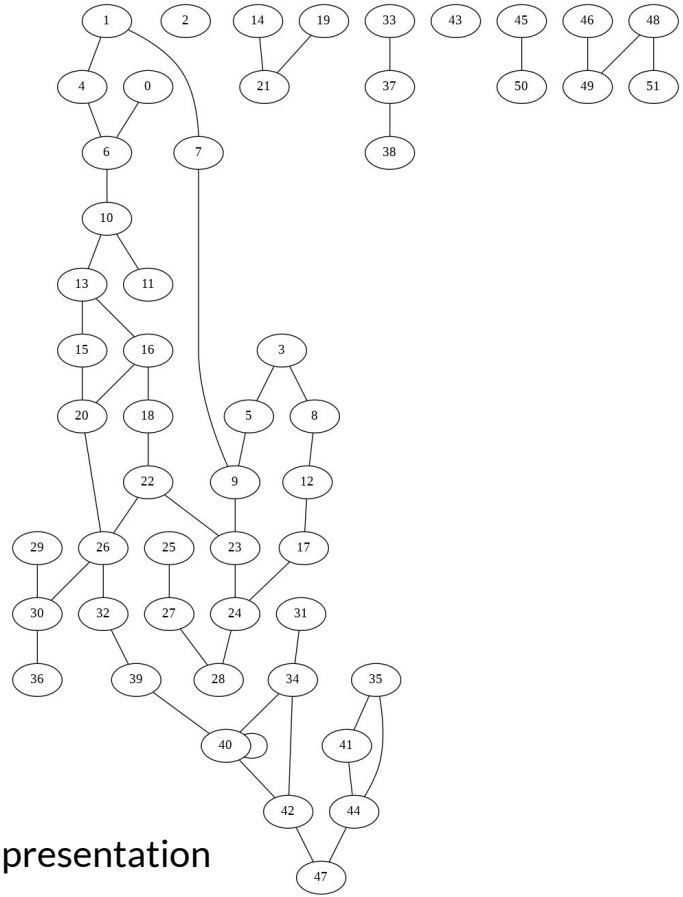


How graph samples are created

Resulting graph



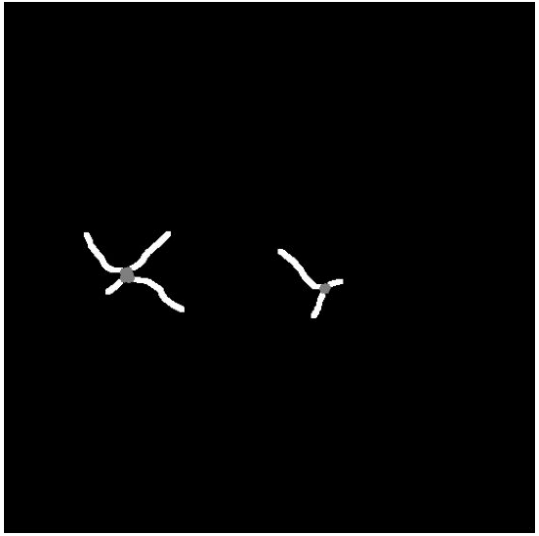
Simplified representation



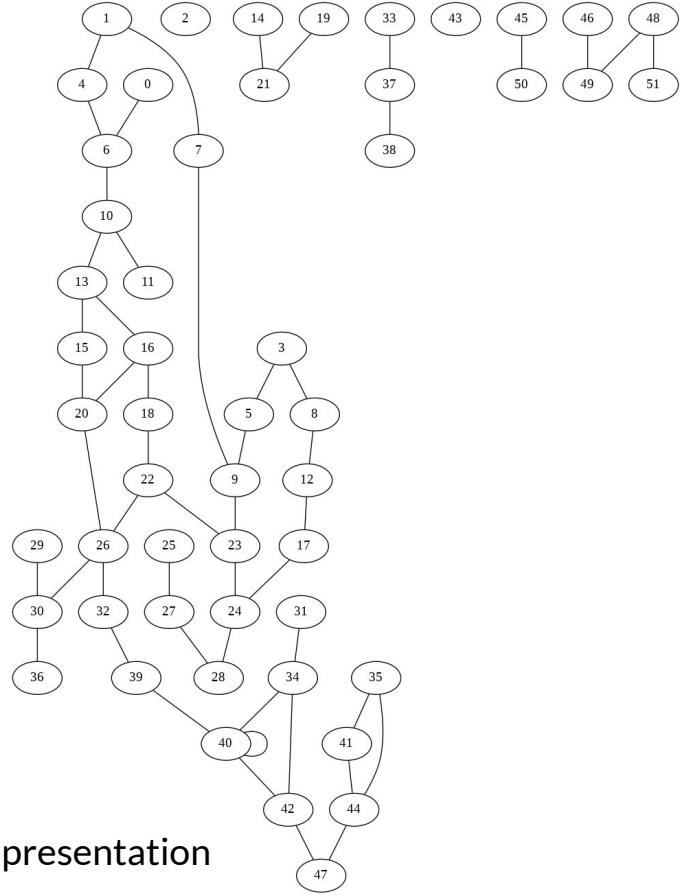
Extended representation

How graph samples are created

Through the labels, features can be traced back to the image: Here node 26 and 28 are plotted back to the respective position in the image, with their edges



Simplified representation



Extended representation

How graph samples are created: Phase 1 - image feature extraction

1- Opening (erosion, then dilation)
2- Node extraction: erosion operation removed the edges and the **following dilation**
expands remaining oval shapes, considered as 'nodes'
3- Skeletonization: original image is skeletonized
4- Removal from the skeletonized image of the locations where nodes have been previously found
5- Dilate skeletonized image:
only edges are now present in the image, this allows to expand them with a dilation, **in order to find a non-null intersection with the previously found nodes**
6- Joints extraction:
Nodes and dilated edges have non null intersection in the image, this allows to find the joints which are necessary to
to which nodes an edge is connected to.
7- Pure Edge extraction: clean edges are the intersection of the regions where nodes == 0 and dilated_edges == 1
8- Pure Node extraction: clean nodes are the intersection of the regions where nodes == 1 and dilated edges == 0

[1] Automatic Extraction of Graph-Like Structures from Binary Imageshttps://www.researchgate.net/publication/220872217_Automatic_Extraction_of_Graph-Like_Structures_from_Binary_Images

How graph samples are created: Phase 2 - abstract graph creation

The opencv function cv2.connectedComponents allows to **find connected components** in images and **labels each of them.** This is done for nodes, edges and joints features.

The procedure returns new images in which each pixel contains the label of the respective connected component.

The key observation of the procedure is that **the intersection between the pixels of the nodes and those of the edges is not empty.**
Therefore, by **observing the label contained in the intersection between the edge image and the node** image, **a mapping of the node labels associated with any given edge can be determined**

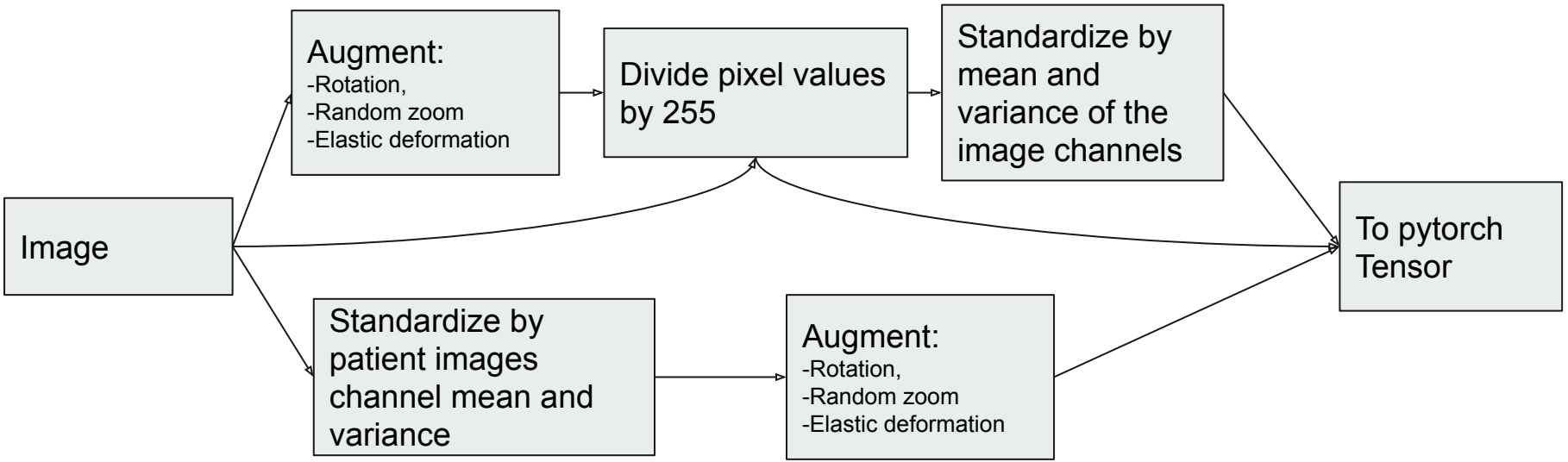
Dataset cleaning

Removal of paired samples (image,mask) that have only one of the two.

Images are resized to 512x512xnum_channels from 2000x2000xnum_channels

Train set masks are thresholded so that the mask only consists of segmented regions(white) and background(black)

Raw samples preprocessing and augmentation pipeline



Preprocessing pipeline that illustrates all possible flows for the experiment configurations

Data augmentation: rotations, random zoom and elastic deformation

Each transformation is applied with the same random parameters to the image and its mask.
Rotation:

- Images are rotated of multiples of 90 degrees

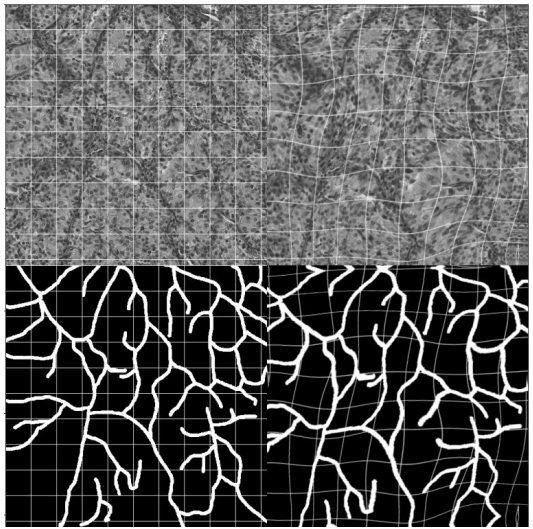
Random zoom

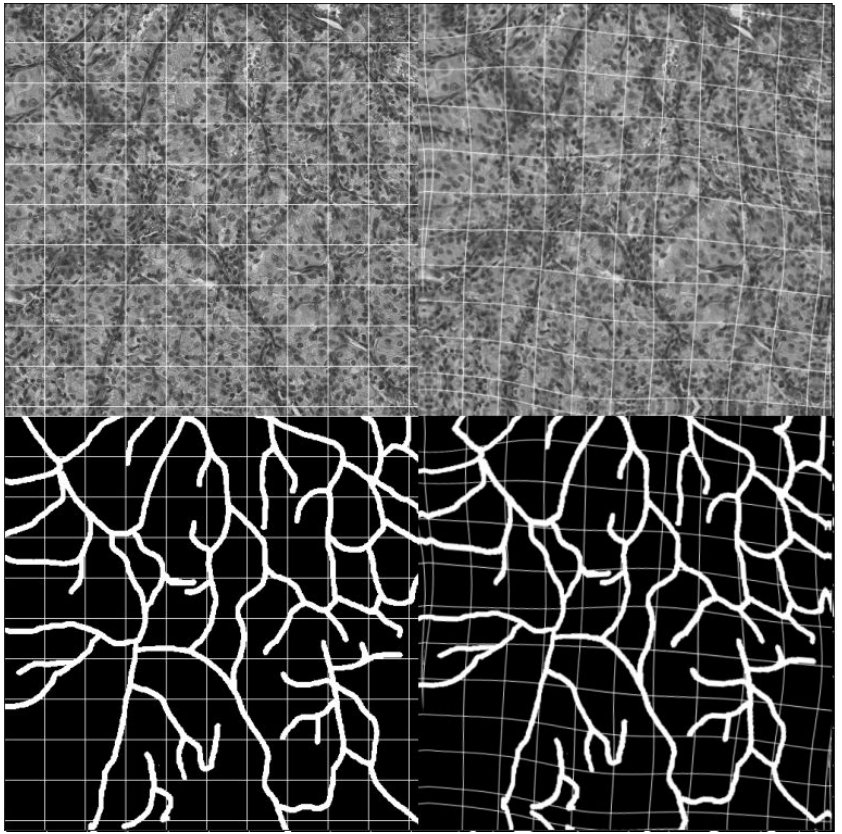
- The implementation crops a square section of the image and then resizes it to 512x512
- The crop consists of a random percentage of the original image shape, between 0.75 and 1.0

Random elastic deformation (Simard 2003 http://cognitivemedium.com/assets/rmnist/Simard.pdf, Adapted implementation provided in: https://www.kaggle.com/bguberfain/elastic-transform-for-data-augmentation)

Elastic deformation

Original image,mask on the left column
Deformed image,mask pair on the right column





The data pipeline for raw images

Preprocessing:

- Resize input to 512x512xn_channels
- (optional) perform data augmentation
- (optional) perform standardization by image channel **or** by patient annotation group image channel
- Permute to channel first format ->n_channelsx512x512
- Create batches of 4 images each

Experiment

- 5-fold cross validation (for CNN and GCN models) or simple train-validation-test split for UNET model
- Train and validate each epoch for n_epochs

The data pipeline for graph samples

Preprocessing:

- Segmented masks (either ground truths or output prediction of the unet model) are processed with the proposed graph extraction pipeline.
- To **each image mask is associated a corresponding extracted abstract graph,** whose nodes and edges are those extracted from the image
- The characteristics of the connected component associated with any given graph node are used as node features:
  - **Width** of the bounding box
  - **Height** of the bounding box
  - **Area** of the connected component
- **Each node feature is standardized** by the train dataset statistics (20% worse accuracy without standardization)
- A graph dataset in which a **sample corresponds to the graph associated with a given segmentation mask** is built

Experiment:

- 5-fold cross validation
- Train of graph convolutional network implemented either **with the torch_geometric library or the stellargraph library**

CNN on raw data

As CNN model, VGG-16 with batchnorm has been selected.

The reason for this, is that interpretability algorithms that will be employed such as **grad-cam** can focus better on regions of interest compared to Resnets.
This has been observed by experimenting with the resulting gradcam plots from trained resnet18 and resnet50.
On the other hand, VGG16 showed much more precise heatmaps.

The hyperparameter configuration is:

- Learning rate in [0.001, 0.0001]
- Batch size = 4
- Epochs in [20, 40, 100]
- Adam optimizer

Unet network

In this work, the Unet has been implemented as in [2].
The only **difference** with respect to[1] is that **no weight map has been used** during training and the **use of padding in order to have an output tensor of equal size to the input.**

This has been described in the paper **to possibly produce artifacts in the segmentation output.**
For simplicity and not needed precision for this task, input and output have been maintained of the same size: the original paper proposed the segmentation of the cancerous region, which naturally demands much higher accuracy.

For this task, **ground truth segmentation mask are much more approximative of the actual vascular network**, so such accuracy is not necessary since this upstream issue is present.

[2] U-Net: Convolutional Networks for Biomedical Image Segmentation 2015 https://arxiv.org/abs/1505.04597

[1] Association of genomic subtypes of lower-grade gliomas with shape features automatically extracted by a deep learning algorithm (https://www.sciencedirect.com/science/article/abs/pii/S0010482519301520?via%3Dihub https://github.com/mateuszbuda/brain-segmentation-pytorch)

Unet network cont'd

Hyperparameters:

- Learning rate in [1e-5, 5e-5, 1e-4]
- Batch size = 4
- Epochs in [20, 40]
- Binary cross entropy loss function (an alternative is Dice loss)
- Adam optimizer

GCN on graph data

The model consists of four graph convolutional layers, each composed of:

- 64 units
- Relu activation

Followed by a global mean pooling operation, dropout and linear layer with weight matrix of shape 64x2, where 2 is the number of classes.

The GCN have been trained in **separate experiments** on **graphs extracted from ground truths and unet segmented predictions.** This has been done as a simple ablation study to assess **whether the features learned from unet can help in extracting useful information from raw images and enhance downstream classification tasks.**

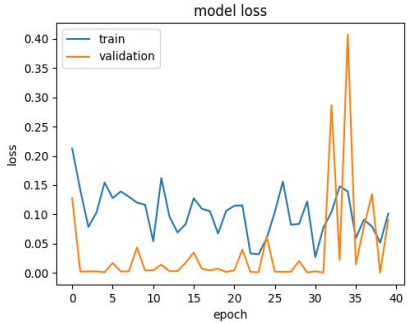
The hyperparameters of the GCN are:

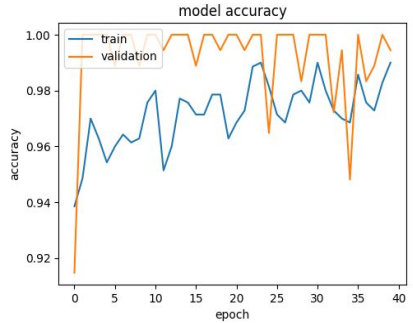
- Learning rate in [1e-4, 1e-3]
- Batch size = 32
- Epochs 100, 200, 400

Experiment results: CNN on raw data

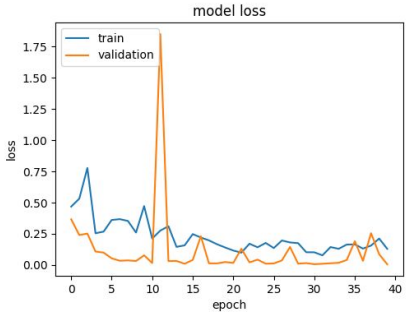
Trained cnn:

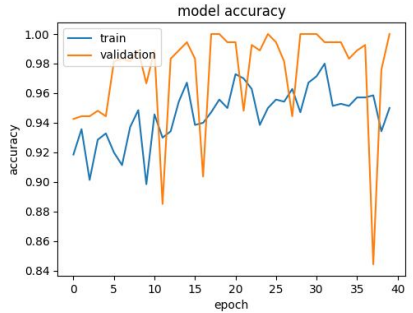
- lr = 0.0001
- RGB images
- Standardization by channel
- Batch size = 4





- Lr = 0.00001
- RGB images
- Standardization by channel
- Batch size = 4



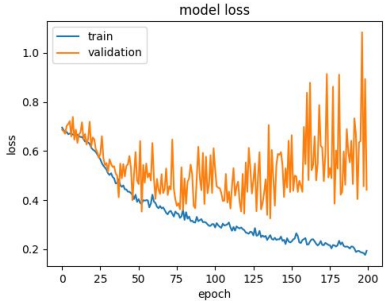


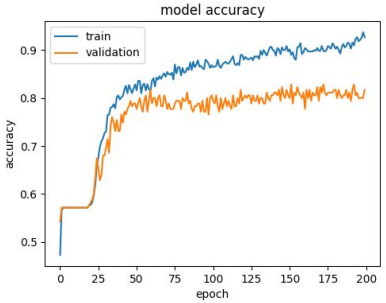
Experiment results: GCN on ground truth graphs

Configuration:

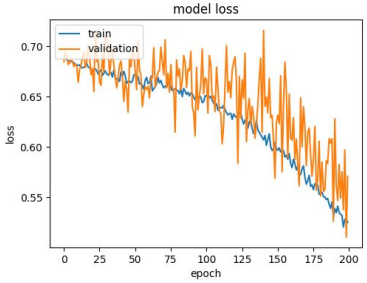
- batch size = 32
- Epochs = 200
- Ground truth dataset

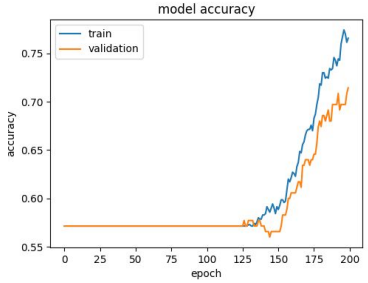
Lr = 1e-3



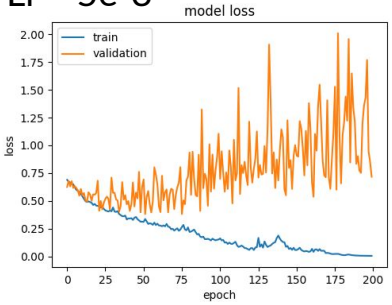


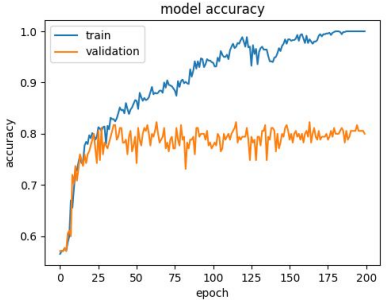
Lr = 1e-4





Lr = 5e-3





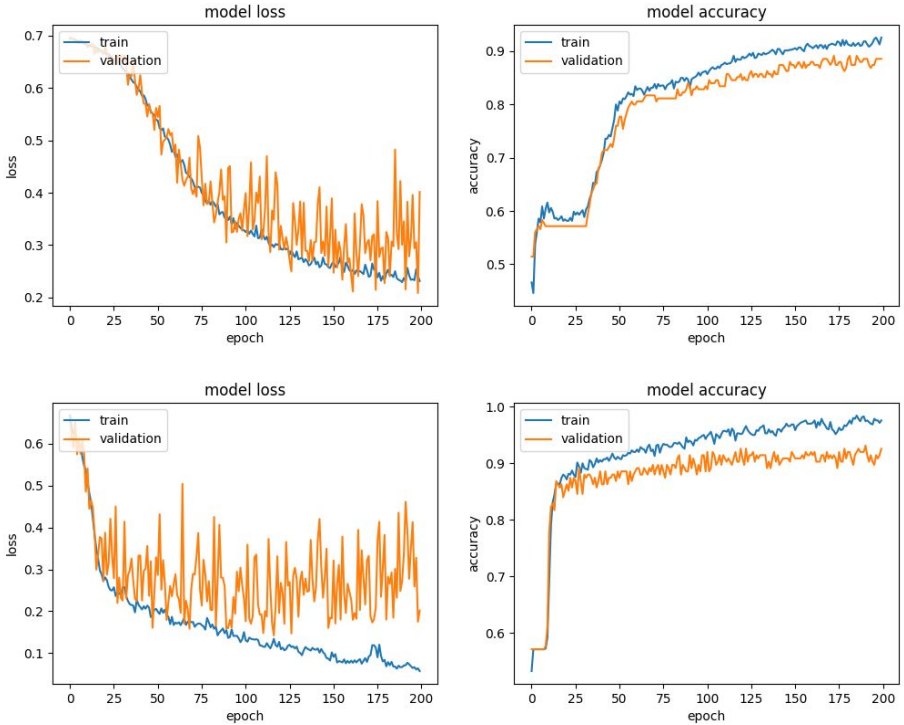
Experiment results:
GCN on unet segmented graphs

Unet:

- Lr = 5e-5
- Batch size =4
- Epochs =40
- RGB images
- Standardization by image channels

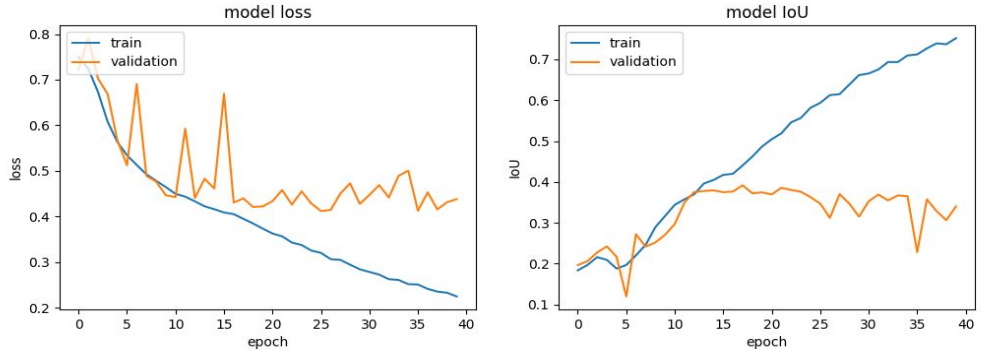
GCN:

- Lr :
  - 0.0001(top)
  - 0.001 (bottom)

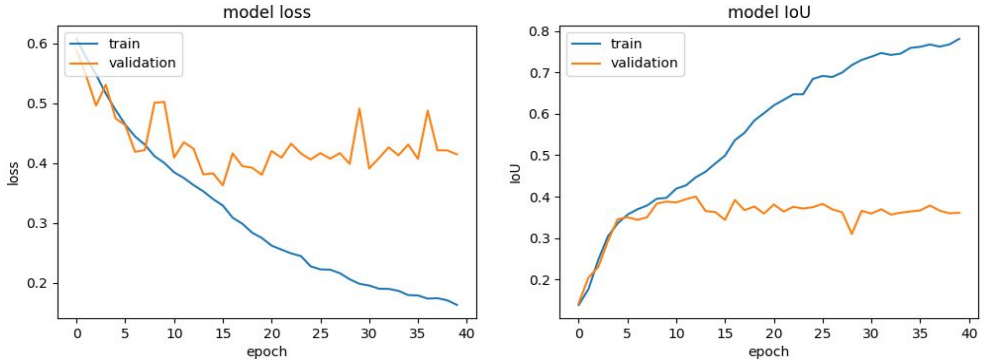


A closer look to the unet segmentation masks

- Lr = 1e-5
- Gray images
- Standardization by channel

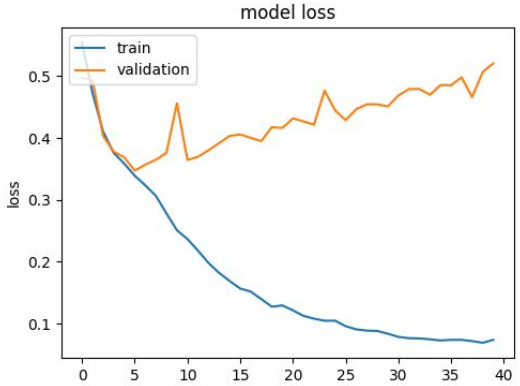


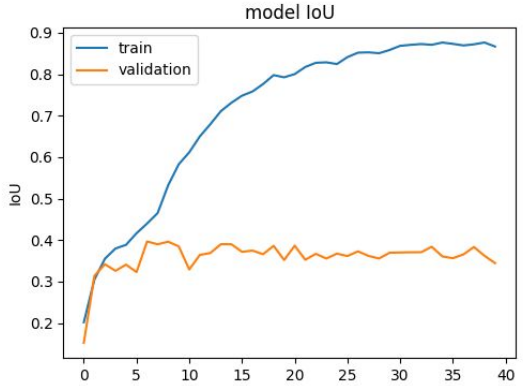
- Lr = 1e-5
- RGB images
- Standardization by channel



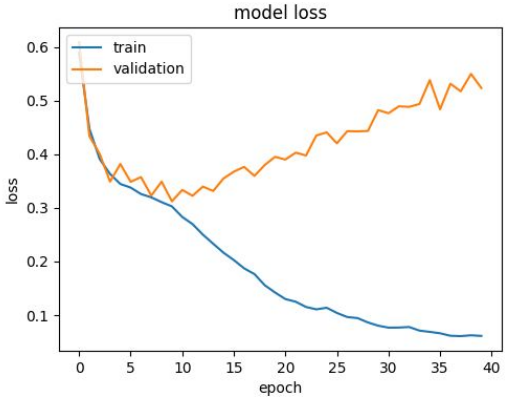
A closer look to the unet segmentation masks

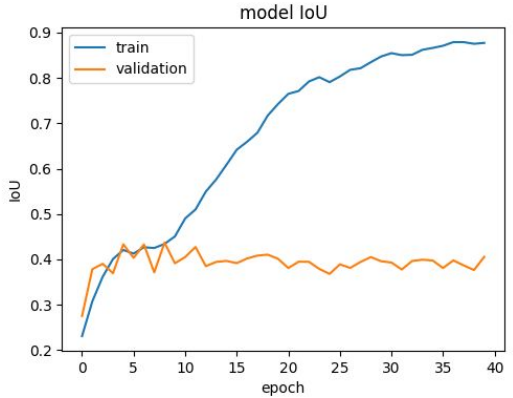
- Lr = 5e-5
- RGB images
- Standardization by channel





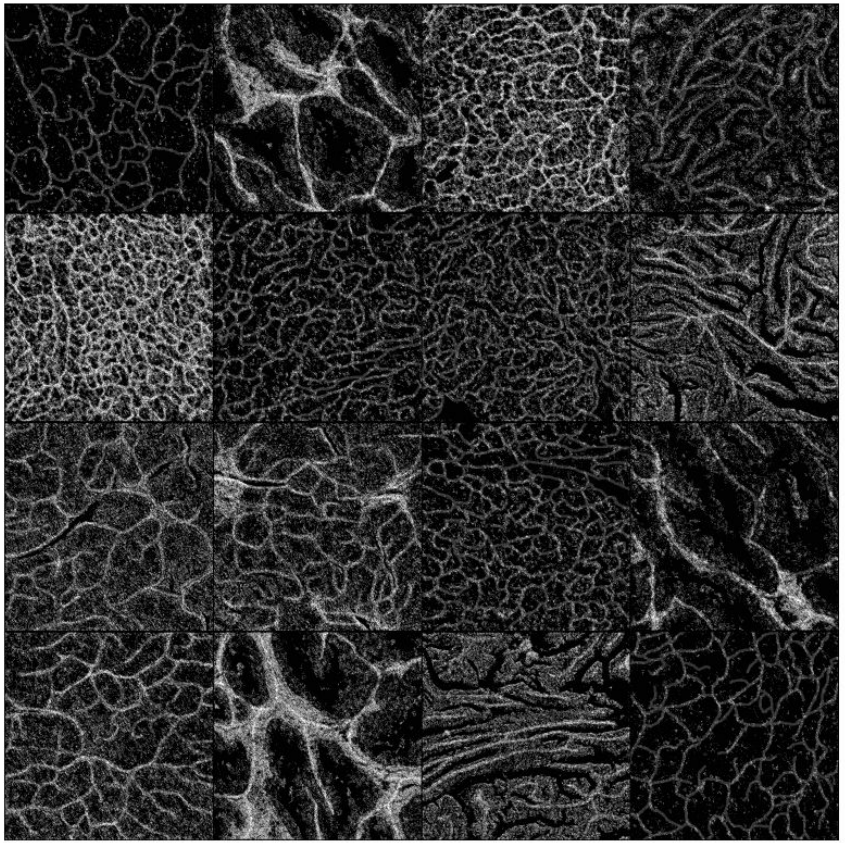
- Lr = 1e-4
- RGB images
- Standardization by channel





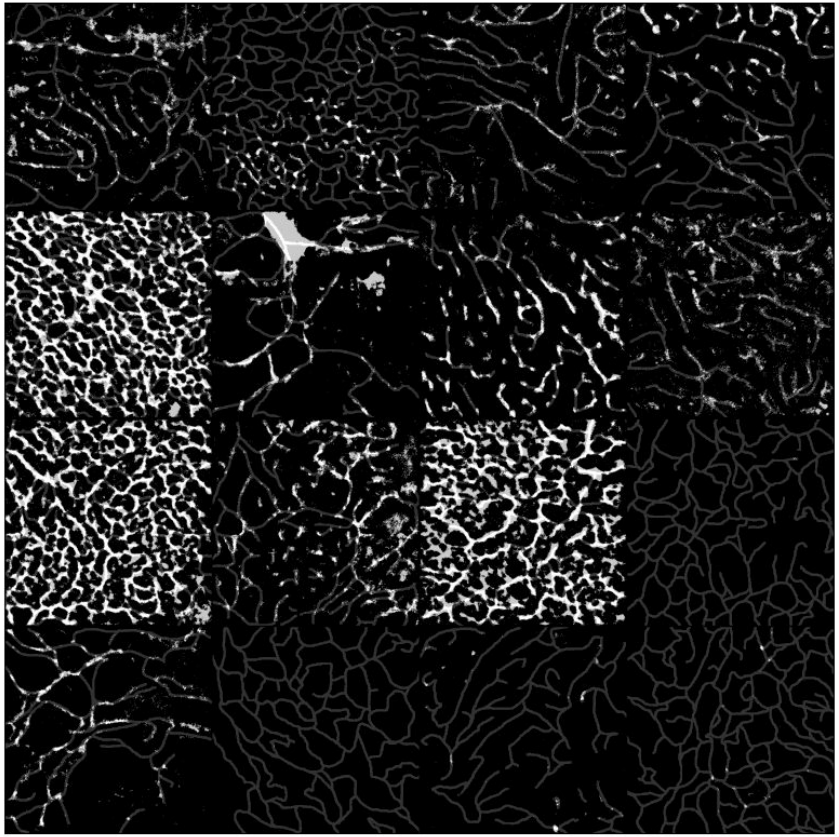
A closer look to the unet segmentation masks

- Lr = 1e-5



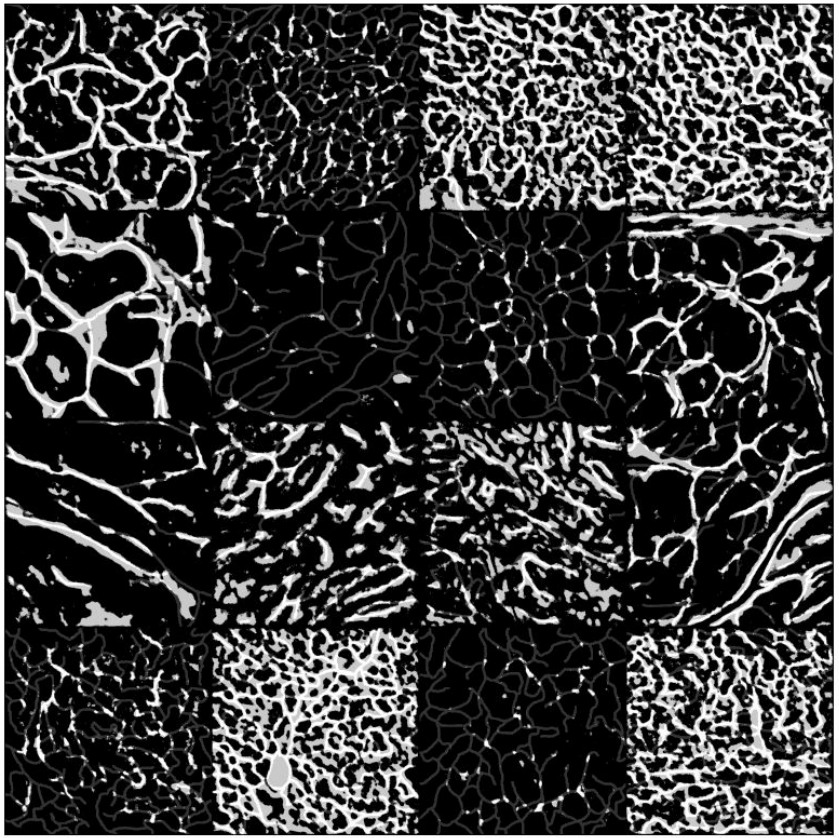
A closer look to the unet segmentation masks

- Lr = 5e-5

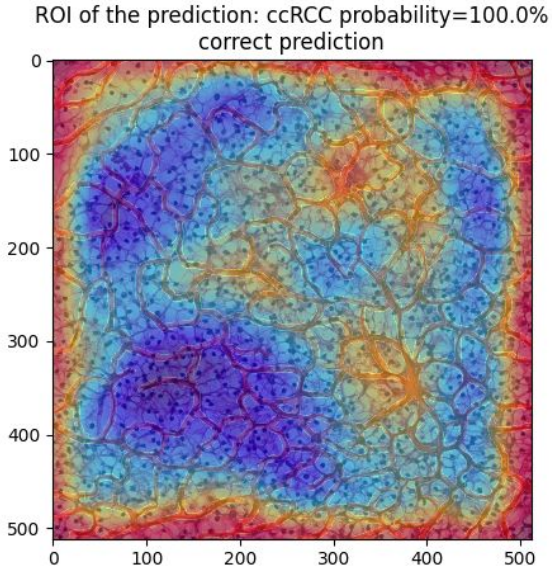


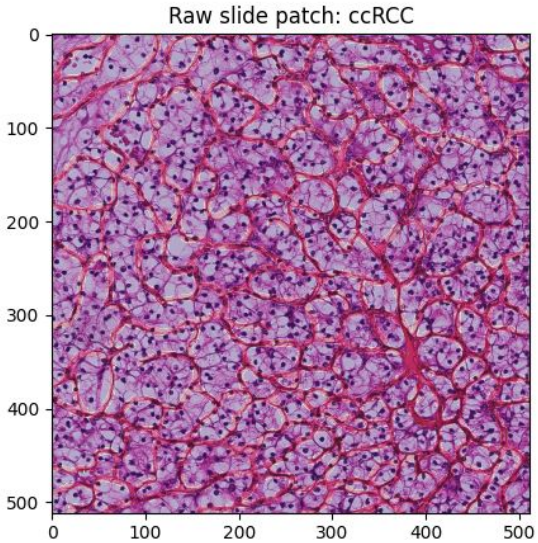
A closer look to the unet segmentation masks

- Lr = 1e-4



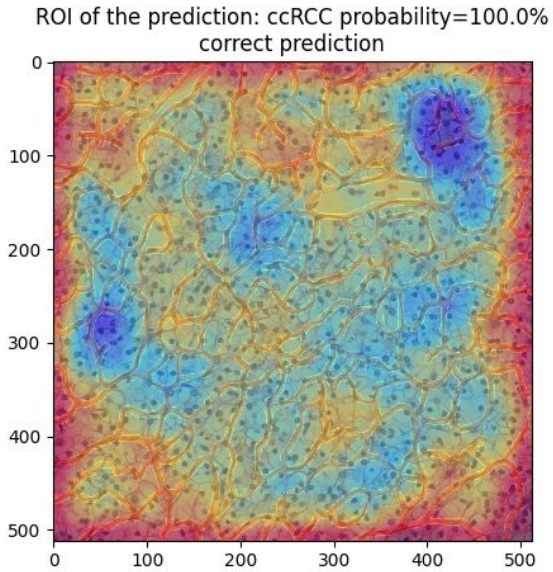
Does the convnet learn graph features?

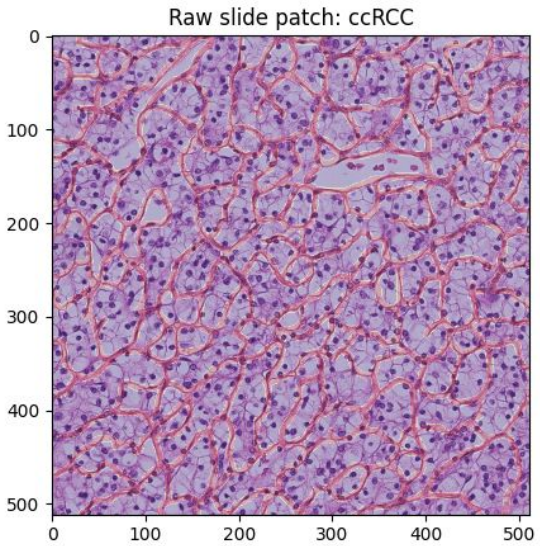




Red defines highly important areas of the image

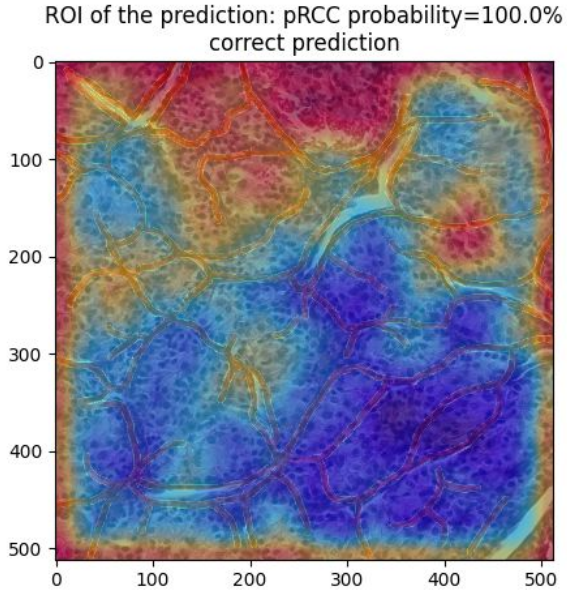
Does the convnet learn graph features?

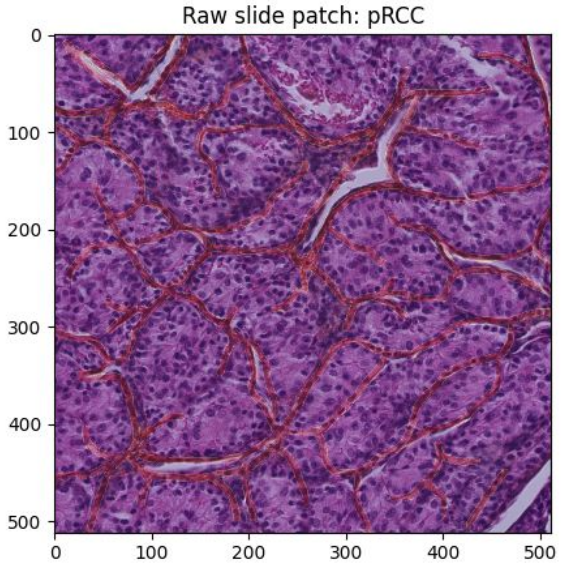




Red defines highly important areas of the image

Does the convnet learn graph features?





Red defines highly important areas of the image

Further experiment:

Removal of cells from raw images before segmenting with unet

Main idea

1. **Cells frequently overlap with the vascular network, as well as the rest of the tissue.**
    a. This factor can easily represent a disturbance in the process of segmentation of the vascular network
    b.
2. Since **no ground truth is available for the cells, an alternative dataset in which they are available is employed.** However the **two domains slightly differ in frequency of cell appearance, colors and scale of the cells:**
    a. Data is **transformed to grayscale**
    b. **Domain adaptation** is employed as a means of finding a common meta representation that is valid for segmenting in the unknown domain(rcc dataset)and in the known domain (the alternative dataset for which cell segmentation labels are available)
    c.
    d. Rcc dataset samples are **split into sub-patches so that the zooming factor of both domains is the same more or less**
    e.
3. **Predict segmentation mask that allows to remove cells in a given image**
4. **Slightly dilate the mask so that cells are completely removed**
5. **Remove cells according to the mask**
6. **Recompose original image from the sub-patches**
7. **Fill the void pixels with the average of the nearest available colors**
8. **Employ the cleaned dataset before proceeding with vascular network segmentation**

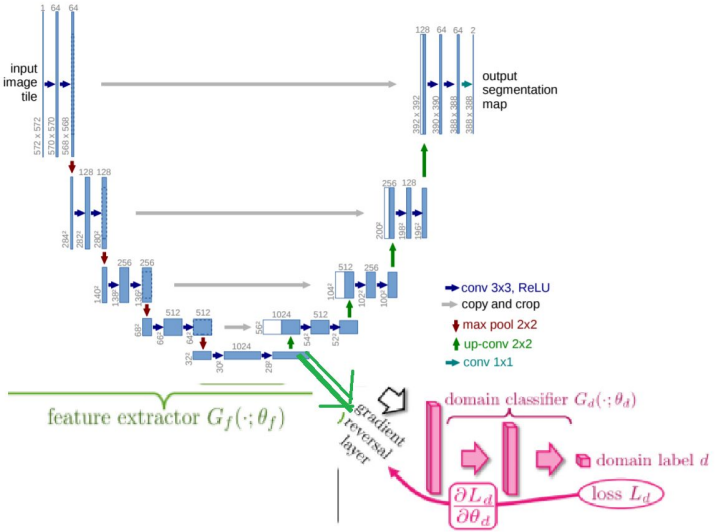
Pipeline changes

- Each original WSI slide patch of size 2000x2000 is resized to 2048x2048 and then **split into 16 squares of equal size(each side of the image is split into 4 parts)**.
- Each new sample is **transformed to gray scale** and color values are **rescaled in the [0., 1.]** range
- **In addition, the alternative dataset**, for which segmentation ground truths are available, is preprocessed in the same way.
- Batch size = 4 is set for both datasets
- The **unet architecture is adapted to satisfy the domain adaptation** setting proposed in (Ganin et al. 2014 Unsupervised Domain Adaptation by Backpropagation):
  - **Unet's contracting and embedding layer before the upsampler** constitute what is defined as **feature extractor in Ganin's paper**
  - The usual **expansive unet layer are kept with the usual connections**
  - The **domain classifier (ganin paper) is attached, together with the gradient reversal layer, to the unet embedding layer (that is in between the contractive and expansive paths)**

Segmentation net architecture

Supervised task aims at segmenting the labelled samples from the alternative dataset

Self supervised task aims at classifying the domain of a given sample (whether it comes from the labelled domain or the rcc dataset which does not have cell segmentation labels)



Segmentation net Training: for each batch

- Get known domain batch, Get unlabelled domain batch

- Compute loss of the Supervised task, with labelled batch (segmentation loss)

- Compute loss of the Self-supervised task with known domain batch (binary crossentropy w. Sigmoid output or crossentropy with softmax output)

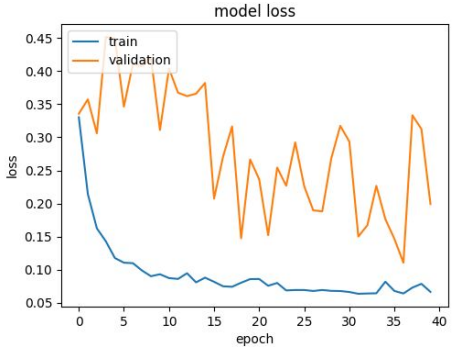
- Compute loss of the Self-supervised task with unlabelled domain batch (binary crossentropy w. Sigmoid output or crossentropy with softmax output)

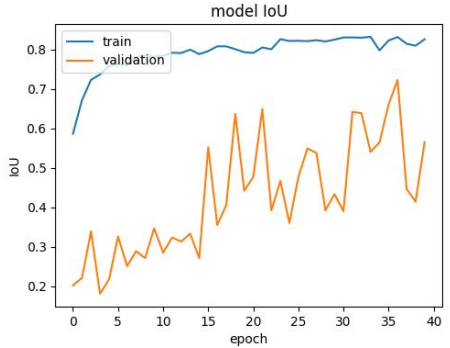
$$Loss = L_{supervised} + \lambda L_{self\_supervised}$$

- Update parameters of the net

Adaptation unet training results

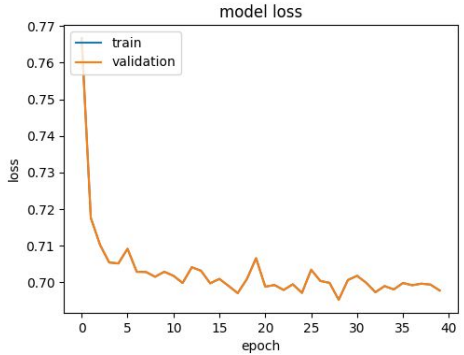
Supervised cell segmentation task results

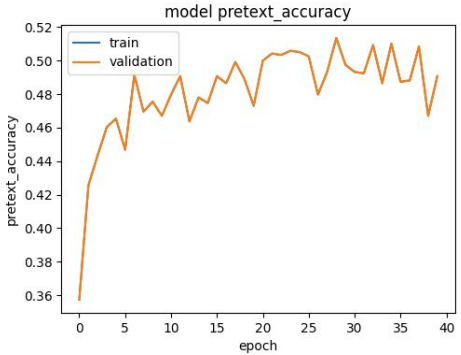




Contemporaneous self-supervised domain classification task results

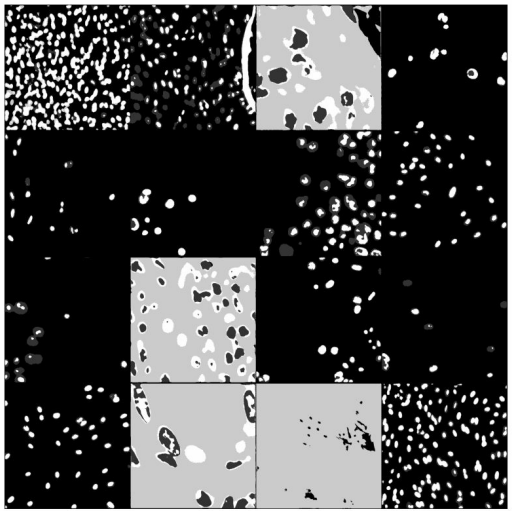
Lr = 0.0001
Batch size = 4 (both batches are of size 4)
Lambda = 1.0
Epochs = 40
Images = grayscale





Adaptation unet cell segmentation (results on labelled dataset)

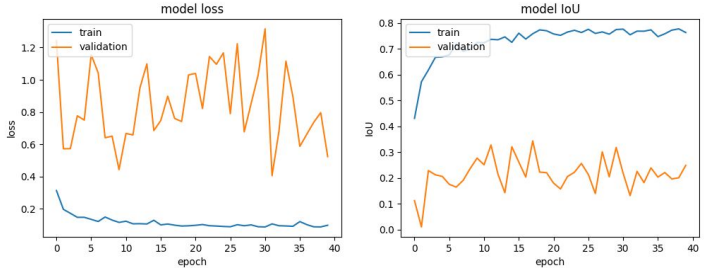
Lr = 0.0001
Batch size = 4 (both batches are of size 4)
Lambda = 1.0
Epochs = 40
Images = grayscale



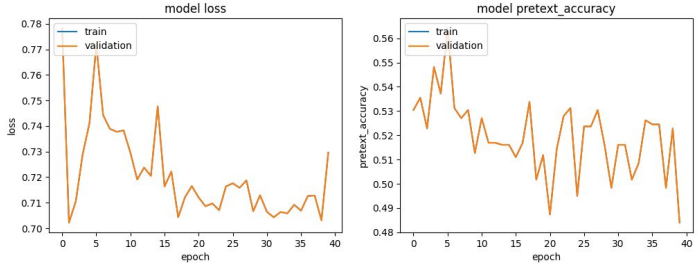
Adaptation unet cell segmentation (results on labelled dataset)

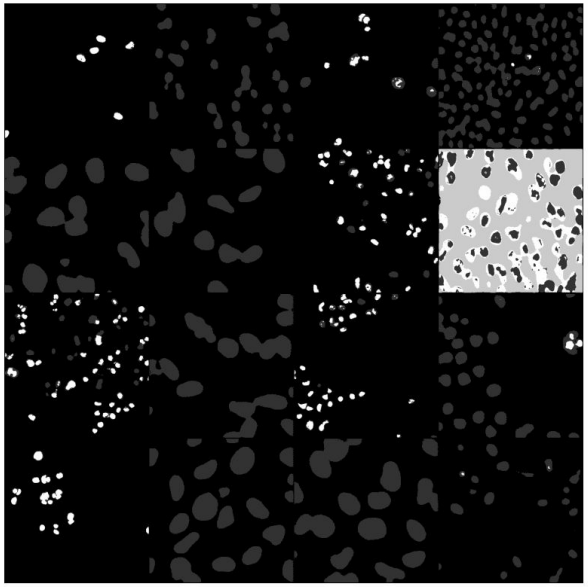
Lr = 0.001
Batch size = 4 (both batches are of size 4)
Lambda = 1.0
Epochs = 40
Images = grayscale

Main task



Pretext task

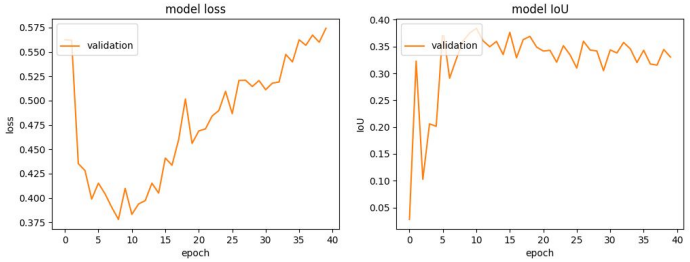


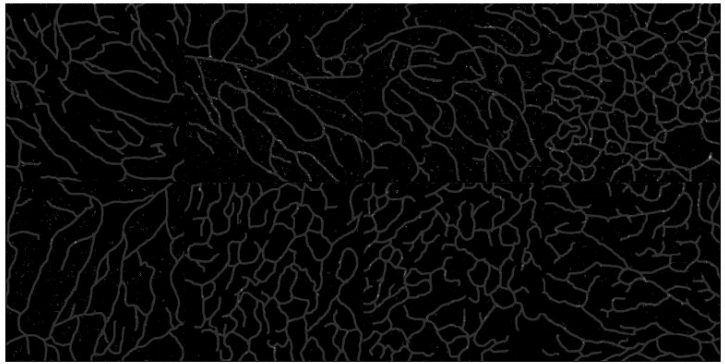


Unet segmentation on cleaned dataset results

Lr = 0.0001
Batch size = 4
Epochs = 40
Images = grayscale

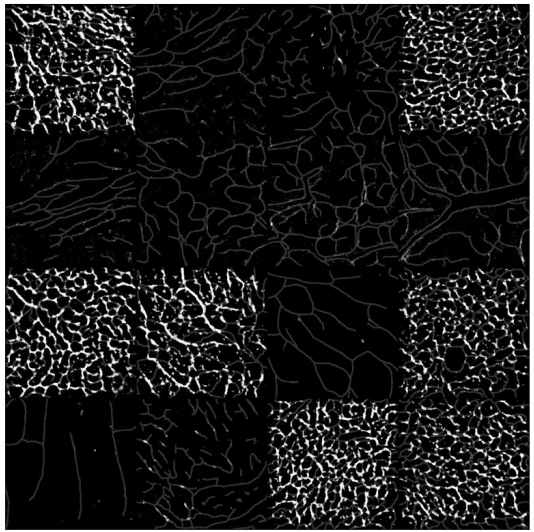
Adaptation unet trained with:
- lr=0.0001
- lambda=1.0



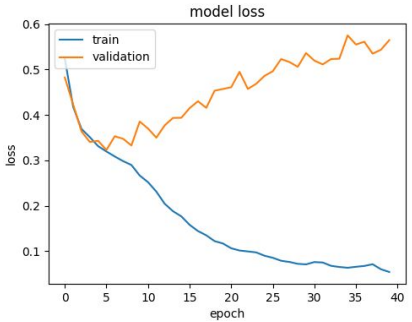


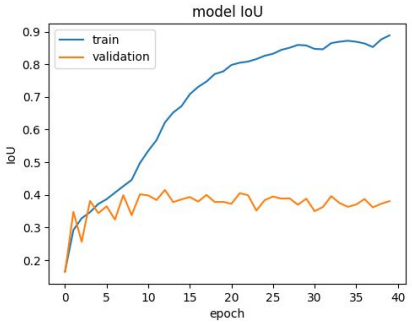
Segmentation progress on the same batch over the epochs(one frame represent a snapshot of an epoch)

Previous method without cell removal (unet on its own)



Lr = 0.0001
Batch size = 4
Images= grayscale
Epochs = 40





Segmentation progress on the same batch over the epochs(one frame represent a snapshot of an epoch)