Image Segmentation

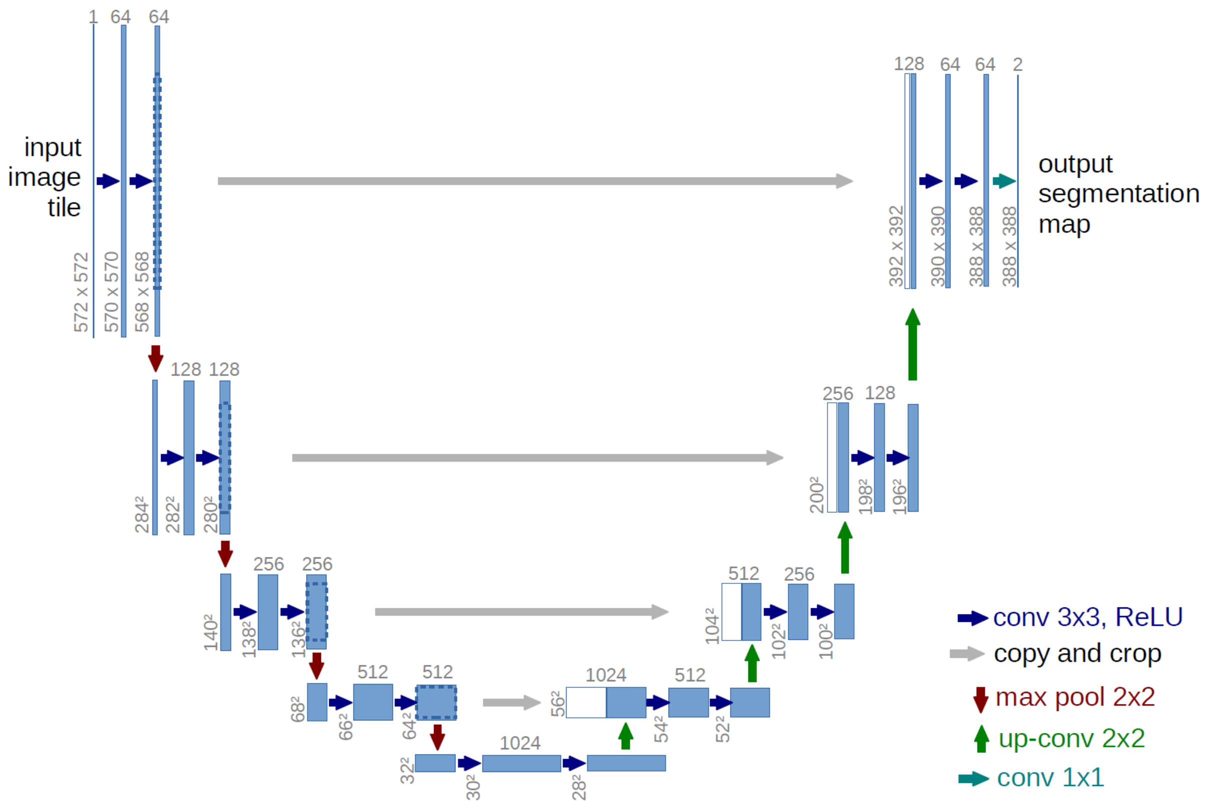
Image segmentation involves dividing an image into multiple segments or regions where each segment corresponds to a meaningful part of the image. The goal is to accurately identify the boundaries of different objects in the image and assign them to their respective segment.

I have chosen the U_Net architecture introduced in "U-Net: Convolutional Network for Biomedical image Segmentation" by Ronneberger et al. for this task.

U-Net architecture consists of two main parts: contraction path and expansion path. The shrinkage path consists of repeated application of convolution layers followed by ReLU activation function and Max pooling operation, that gradually reduce the spatial dimensions of the feature maps. This path is designed to capture the context of the input image.

The expanding path is a mirror image of the shrinking path, with up convolution layers that gradually increase the spatial dimensions of the feature maps. This path is designed in such a way that it enables accurate positioning of objects in the image.

The U-Net architecture also includes jump connections that connect the respective layers in the contraction and expansion paths. These connections help preserve high-resolution information from the shrinking path and combine it with background information from the expanding path, resulting in more accurate segmentation masks.

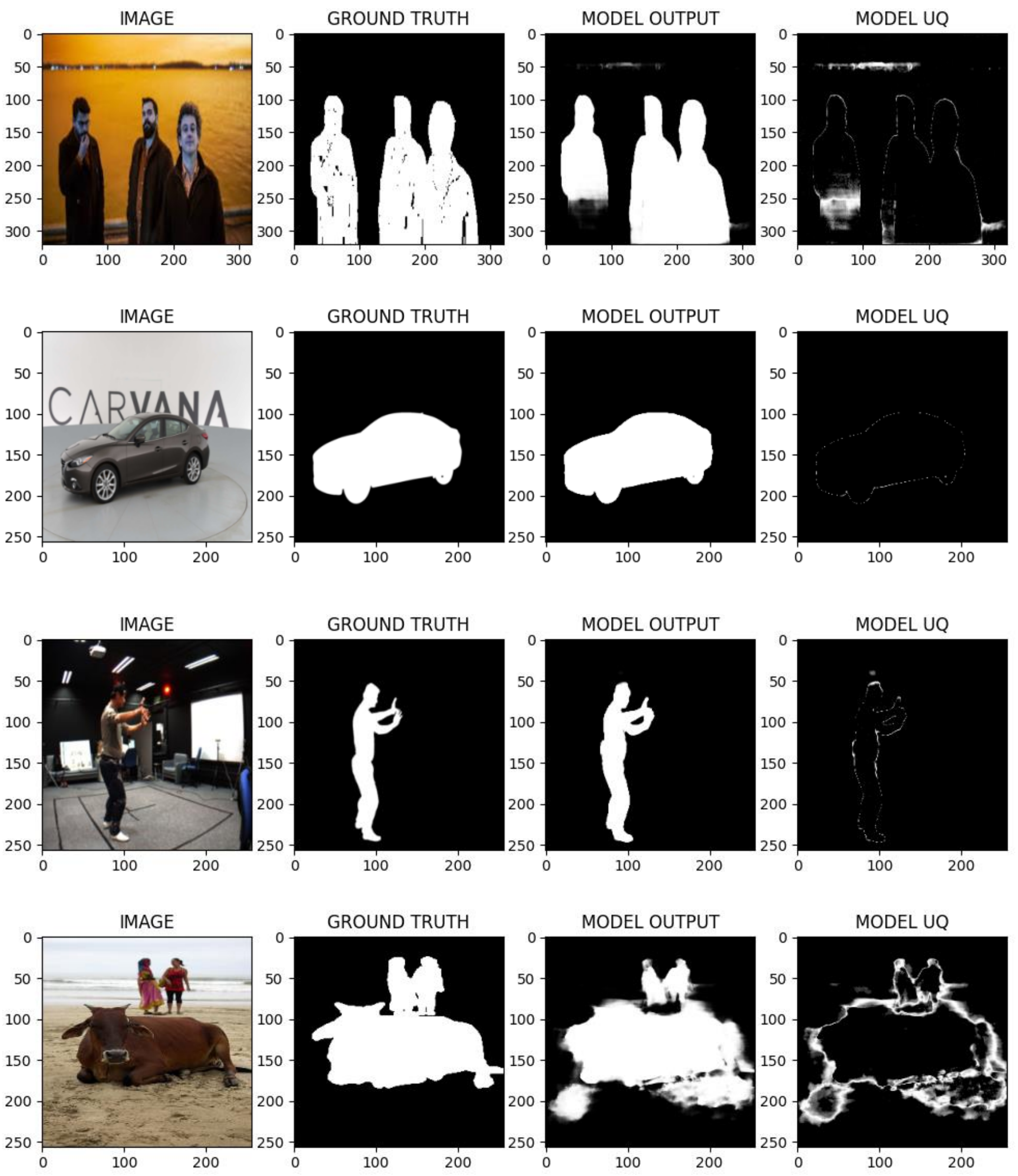


Dataset:

In this project I have used 4 different datasets:

- Custom dataset for human segmentation: A dataset with 300 images of humans with some background and a corresponding binary mask for each of these images.
- Carvana image masking challenge dataset: This dataset contains 5088 of car images. Each car has exactly 16 images, each one taken at different angles.
- MADS (Martial Arts, Dancing and Sports) dataset comprises fast and complex activities. It has been published for the task of estimating human posture. However, before determining the human pose, the person needs to be detected as a segment in the video. This dataset consists of 1192 images and masks.
- PASCAL Visual Object Classes (VOC) 2012 dataset: only used in binary segmentation.

Here is a sample of prediction for each dataset respectively (can be found in output folder of the project):



Project Architecture:

The project consists of the following folders and files:

- conf:
  - `config.yaml` : Includes all the changeable parameters used in running experiments
- data: After running `train.py` the selected dataset will be downloaded and saved in this folder.
- docker: contains docker file.
- segment:
  - `analytics.py:` Includes code for the uncertainty analytics, metrics, histogram.
  - `datasets.py:` Includes classes for retrieving and preparing each dataset.
  - `helper.py:` Includes plot functions.
  - `inference.py:` Sets up the inference configurations.
  - `nn_models.py:` Include U-Net architecture.
- predict.py: Generate predicted masks with standard-deviation.
- train.py: Includes parameters initializations, data loading, training loop, etc.

Note:

- All training and prediction parameters are configured with the `conf/base.yaml` file.
- After activating the conda environment with `conda activate segment_uq` (or using the docker image), train the network with python `train.py` and generate predicted masks with standard-deviation estimates by python `predict.py`.

Uncertainty Quantification:

Monte Carlo dropout is used to estimate model uncertainty, which means a dropout with dropout rate of 50% has been added after every convolutional layer.

At test time, multiple predictions of the same input have been obtained by running the model several times (e.g., 100 evaluations), due to dropout each inference will produce a slightly different result. Final mask prediction and uncertainty is simply the mean and standard deviation of the stochastic predictions for each input image.

I propose 3 metrics to quantify the uncertainty calculated by the standard deviation matrix (i.e., matrix of pixels) per image in the dataset; therefore, following metrics are applied to the standard deviation matrix for each image:

1. Frobenius norm, average of squares of pixel values in the matrix. This metric measures an average uncertainty over all pixels, smoothing the overall estimate.
2. L1-norm, maximum absolute pixel-value in the matrix. This is sensitive to outlier (most uncertain) pixels, so it is the most conservative (strictest) metric for evaluating uncertainty.
3. Maximum singular value (i.e., SVD decomposition) of the standard deviation matrix. This metric is a measure of the uncertainty contribution (singular value) from the strongest mode (eigenvector) present in the standard deviation matrix. Not sensitive to individual pixels (unlike L1-norm) and not considering all pixels equally (unlike Frobenius norm), this metric measures a subset of pixels that contribute the most to the overall uncertainty.

Afterwards, for each set of training configuration parameters, I computed the histogram of the evaluated 3 metrics per image, then I calculate the median value of each metric over the entire dataset that provides 3 scalar values to quantify the overall uncertainty for each set of parameters.

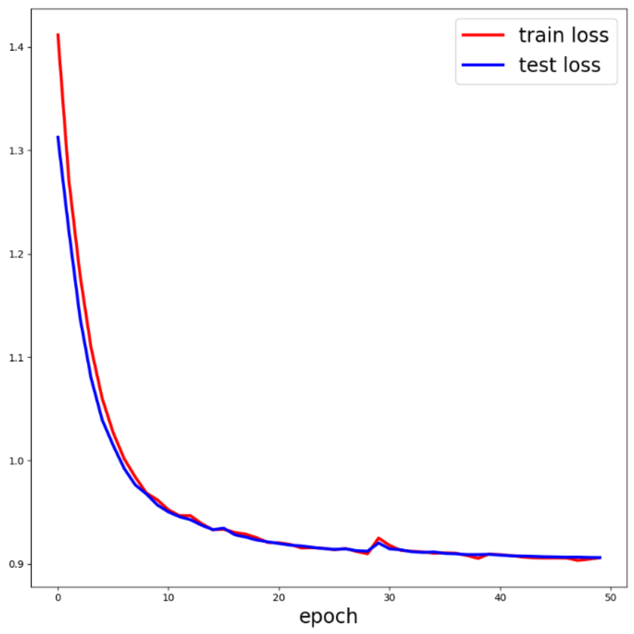
Note that for each metric, the least quantified uncertainty is measured by the lowest median values across the dataset.

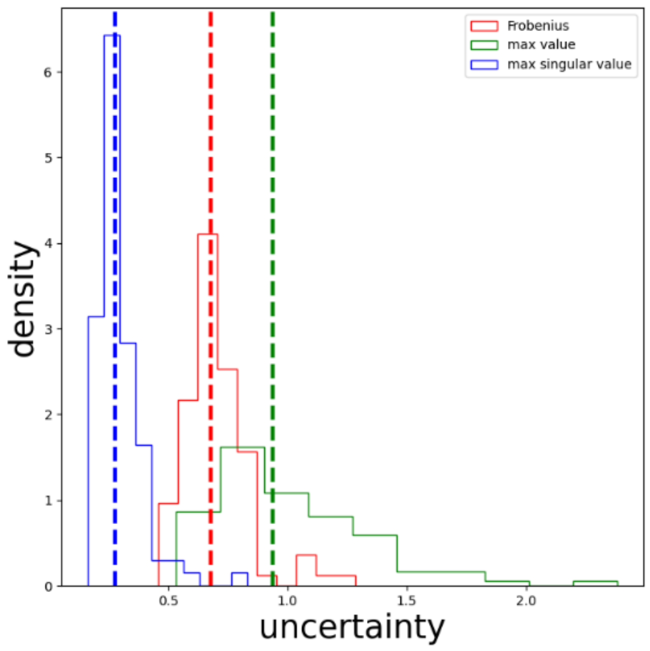
Note:

- MADS dataset has been used for generating uncertainty experiments because it is optimized in terms of size and complexity among all mentioned datasets for the purpose of binary segmentation.

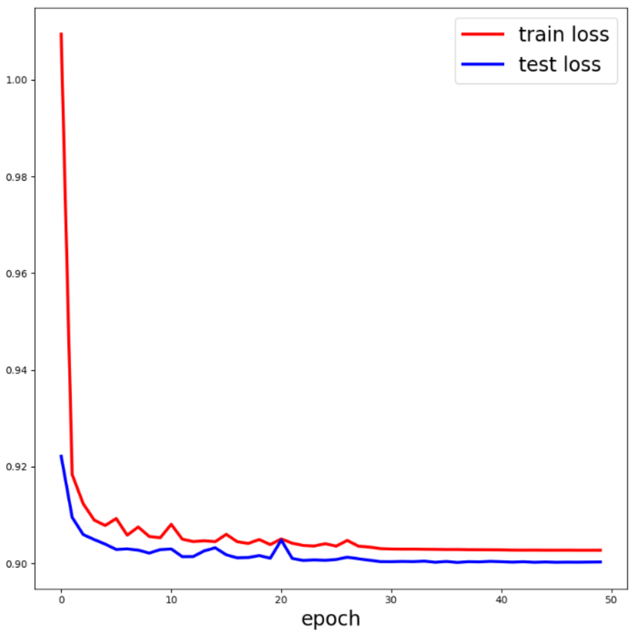
Effect of batch size: (Fine Tuning parameters)

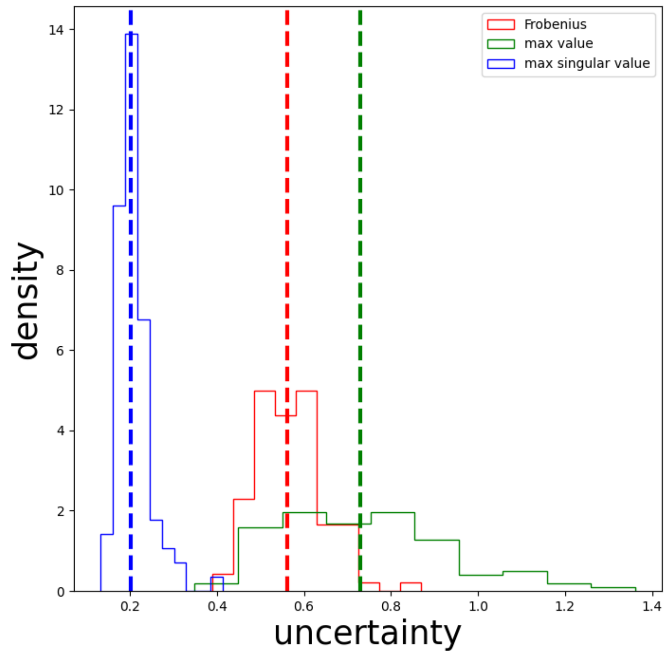
- Batch size: 32, Optimizer: Adam, Learning rate:1e-3, U-Net base exp: 5, Loss function: BCE+Dice





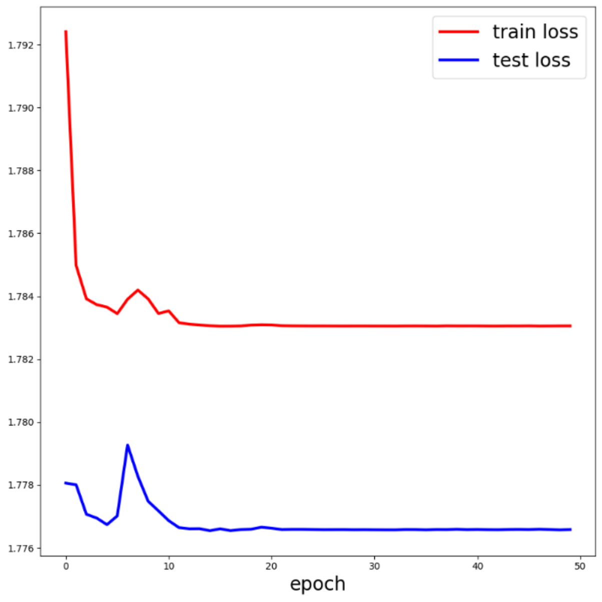
- Batch size: 1, Optimizer: Adam, Learning rate:1e-3, U-Net base exp: 5, Loss function: BCE+Dice

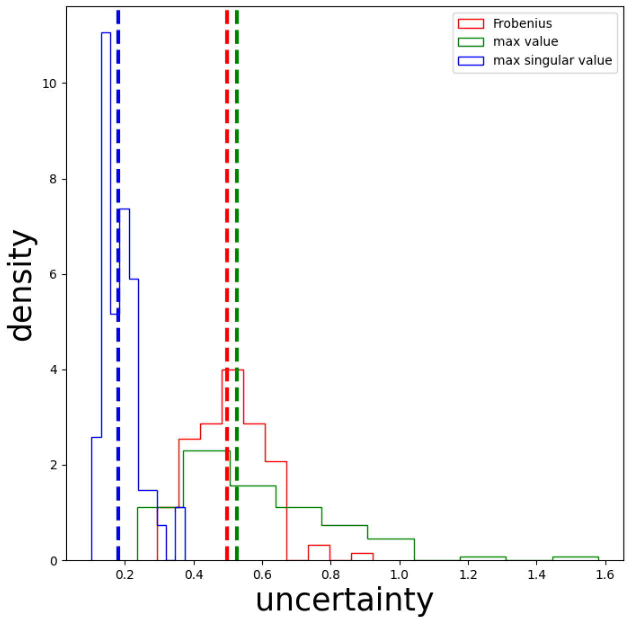




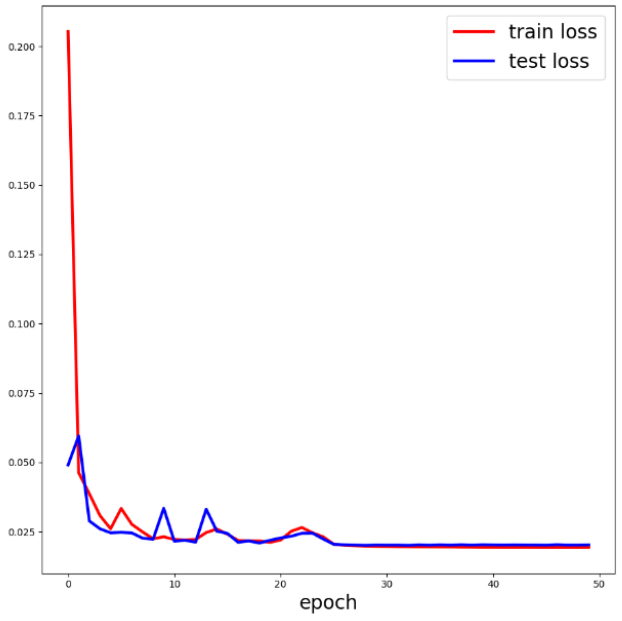
Effect of loss function:

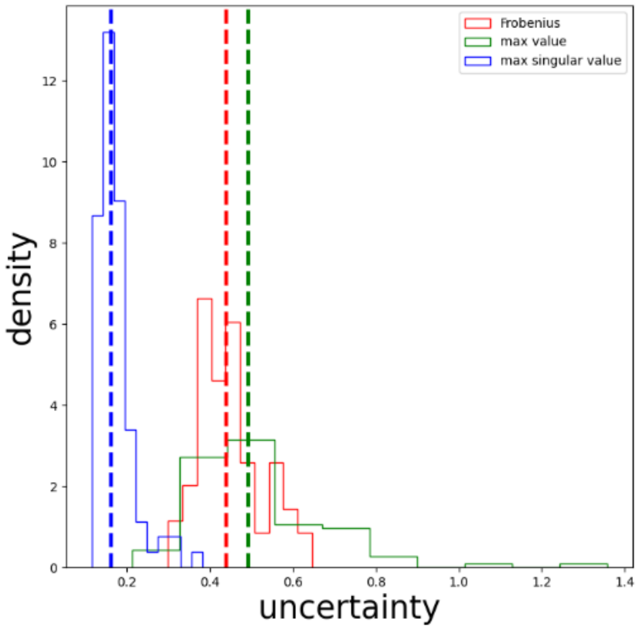
- Batch size: 1, Optimizer: Adam, Learning rate:1e-3, U-Net base exp: 5, Loss function: Dice





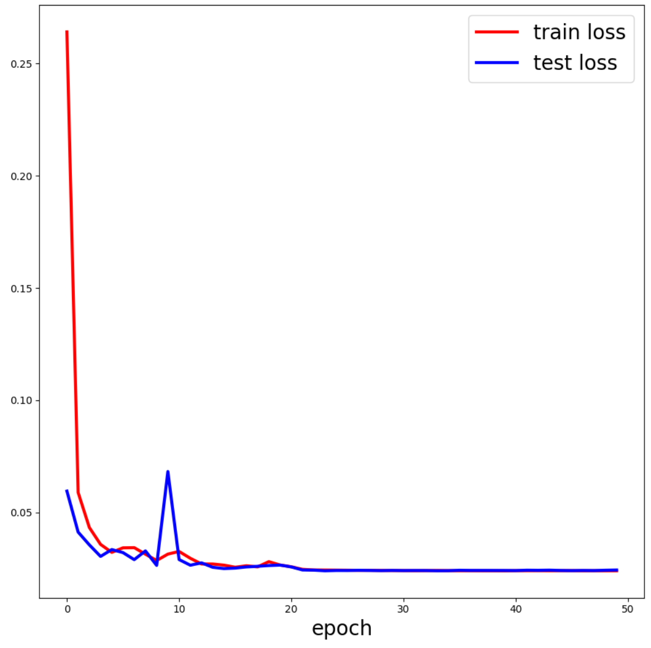
- Batch size: 1, Optimizer: Adam, Learning rate:1e-3, U-Net base exp: 5, Loss function: BCE

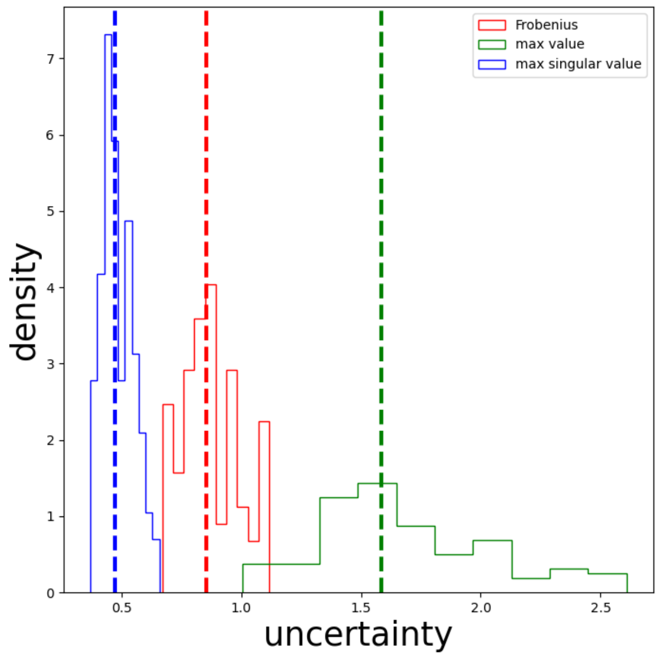




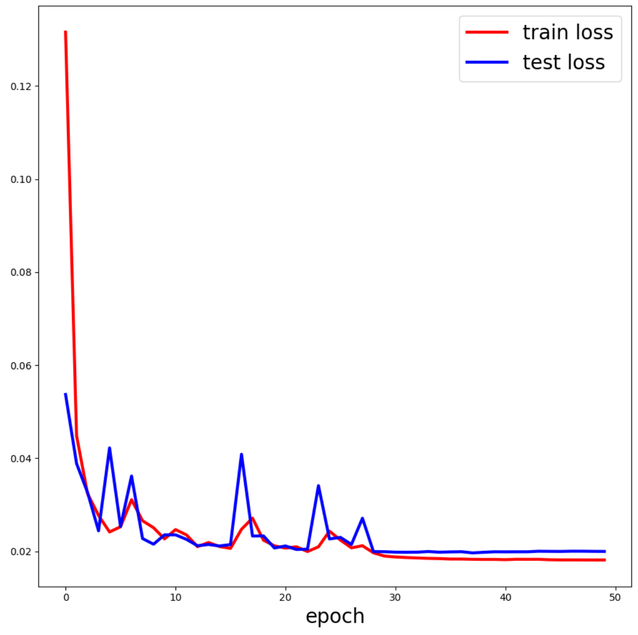
Effect of U-Net blocks:

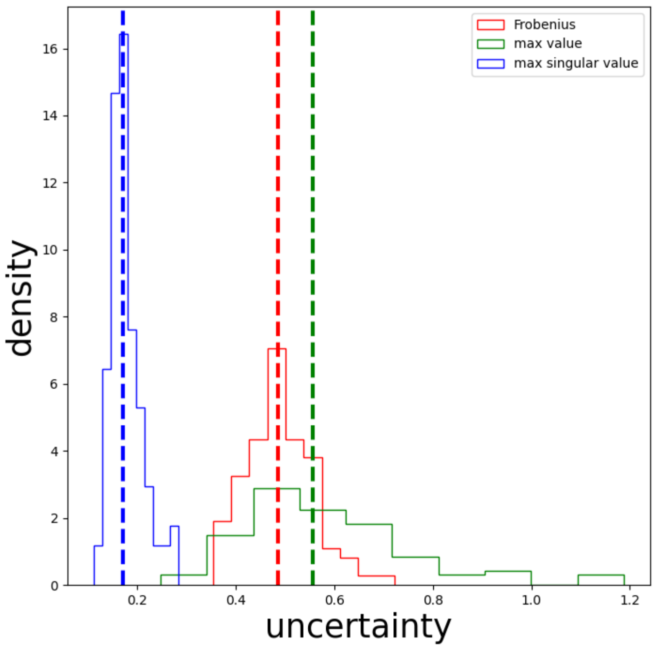
- Batch size: 1, Optimizer: Adam, Learning rate:1e-3, U-Net base exp: 4, Loss function: BCE



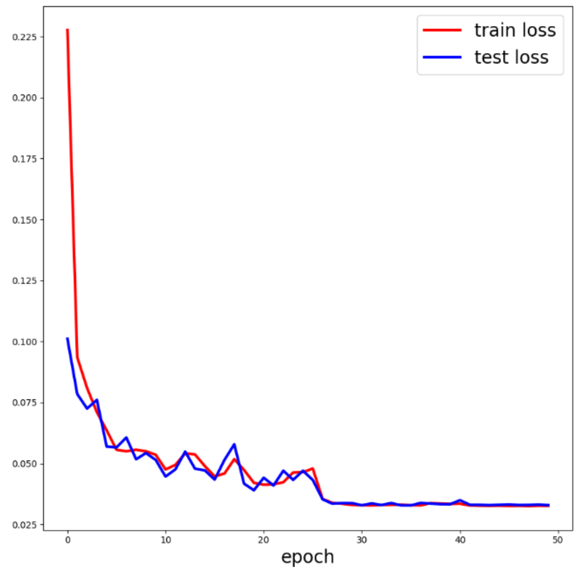


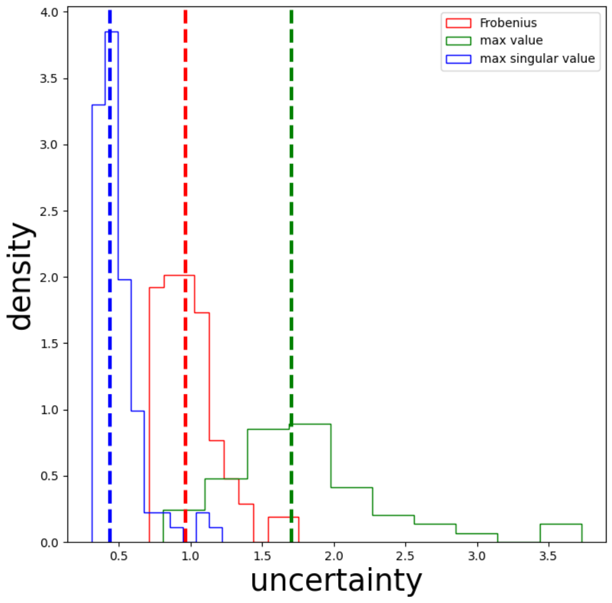
- Batch size: 1, Optimizer: Adam, Learning rate:1e-3, U-Net base exp: 6, Loss function: BCE





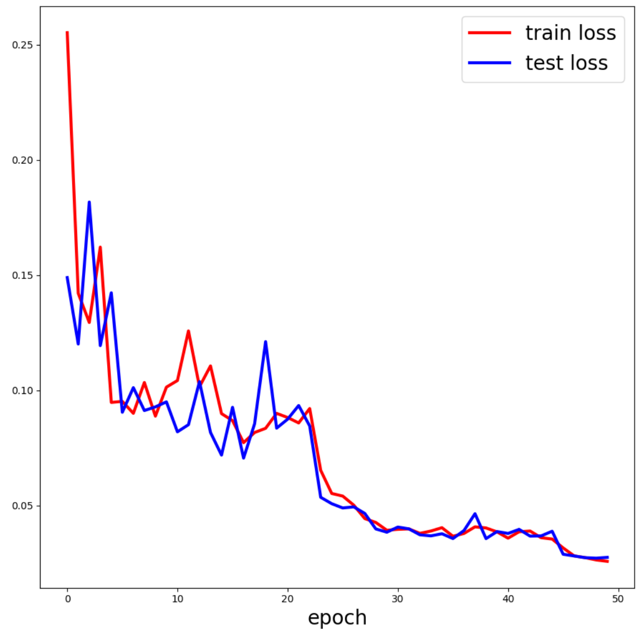
Effect of Activation function: (this item is not included in config.yaml, sigmoid used instead of ReLU)

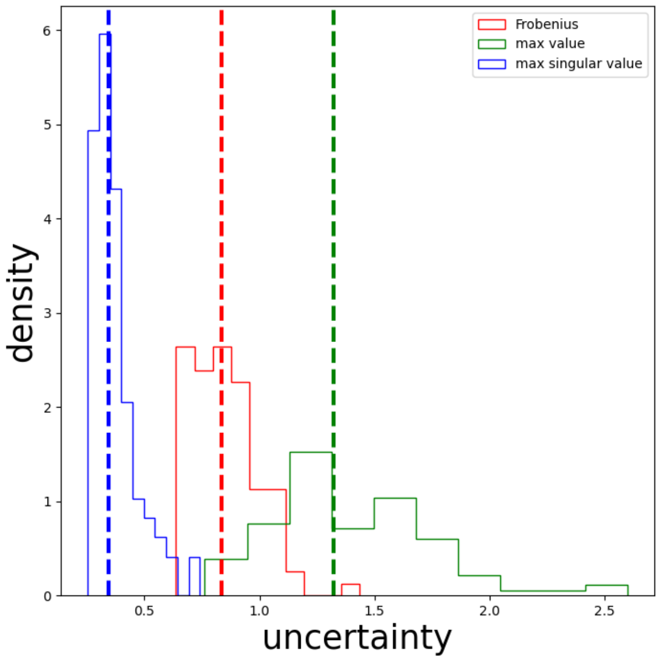




Effect of Optimizer:

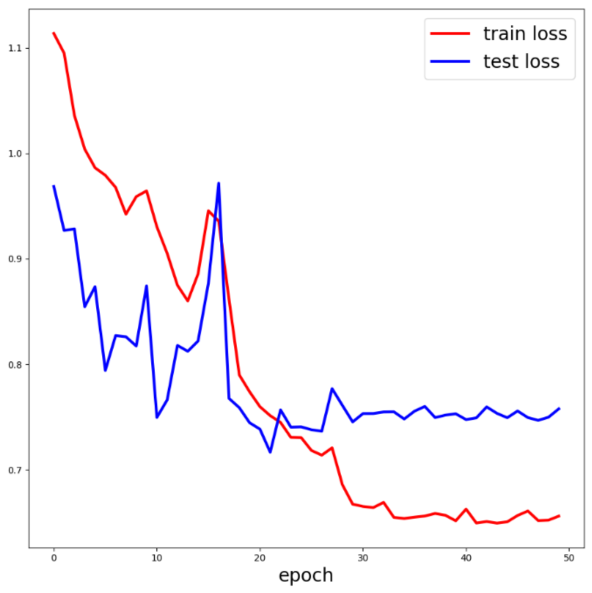
- Batch size: 1, Optimizer: Rmsprop, Learning rate:1e-4, U-Net base exp: 5, Loss function: BCE

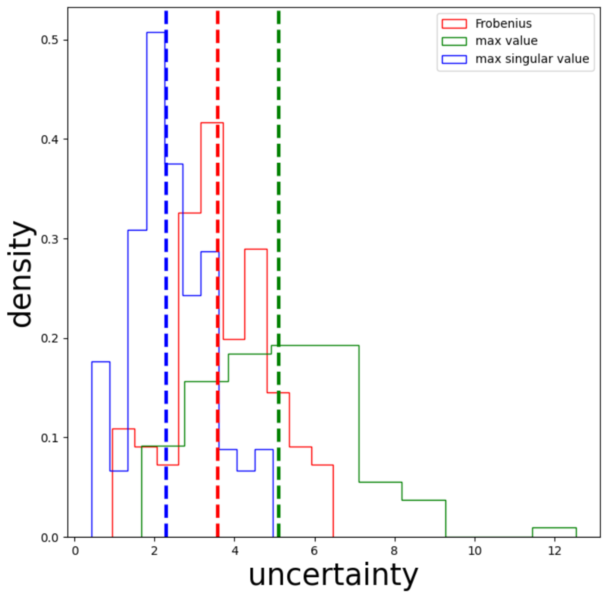




Custom Dataset:

- Batch size: 1, Optimizer: Adam, Learning rate:1e-3, U-Net base exp: 5, Loss function: BCE





Future work & Improvements:

- Extend the code for multiclass image segmentation.
- Change the model architecture with other variants of the U-net