A deep learning approach for predicting outcomes of triple-negative breast cancer

Student:
Arvid Larsson
ar7144la-s@student.lu.se

Supervisor:
Mikael Nilsson
mikael.nilsson@math.lth.se

Deputy supervisor:
Emma Niméus
emma.nimeus@med.lu.se

Examiner:
Kalle Åström
kalle@maths.lth.se

January 21, 2021

Abstract

Breast cancer is the most common cancer in women. Triple-negative breast cancer affects 10-20% of breast cancer patients and is associated with an especially bad prognosis. Today, tissue slides are assessed manually by a clinician to set a prognosis. However, the prediction of outcomes could possibly be improved using machine learning. This work investigates various machine learning techniques for the task. In particular, a U-Net was trained on public data and was used to detect cells in microscopic images of H&E-stained triple-negative breast cancer tissue. The detected cells were then classified using logistic regression. The detected cells served as a basis for the prediction of local relapse, distant relapse, and overall fatality in a cohort of 155 patients diagnosed with triple-negative breast cancer. The accuracy of models fitted to features extracted using machine learning was compared to the accuracy of models fitted to features estimated by a clinician. The features extracted using machine learning was found to yield as good, or better, predictions compared to estimated features. A high number of tumor-infiltrating lymphocytes was associated with a better prognosis. This shows that machine learning can be used to find biomarkers in microscopic tissue images and use these to predict outcomes in cancer patients. The source code used in this work is published on Github: `https://github.com/karvla/histosnet`

Acknowledgements

This work would not have been possible without the help I've gotten from friends and experts. I want to thank my supervisors Mikael Nilsson at the Center of Mathematics and Emma Niméus at Clinical Sciences, Department of Surgery. Mikael gave me excellent advice regarding machine learning, statistics, and scientific writing. Emma was always available to answer my questions about the state of breast cancer diagnostics and medicine. Emma's expertise did more than make up for my lack of domain knowled ge. Felicia Leion at Biomarkers and epidemiology painstakingly annotated the contours and the type of thousands of cells from the TNBC-cohort. These annotations are of excellent quality and training the cell classifier would not have been possible without them. For this, and for answering my histology related questions, I am truly grateful. Felicia also made the stromal TILs estimates for the TMA images. Johan Hartman made the estimates for the WSIs.

I also want to thank Patrik Laurell and Dwight Lidman for laying the foundation of the first version of the cell detector. This sped up my progress in the early stage of this project.

Thanks to Johannes Kumra and the department at Quantitative infection biology for letting me use their GPU-equipped server for development and model-training, and thanks to BioMS at Lund University and all its personnel for letting me work at their facilities.

I want to thank Erik Hartman and Jonas Wisbrant for proofreading. I also want to thank Patrik Nilsson for keeping me company through video calls while working from home.

Contents

1 Introduction

Breast cancer is the second leading cause of cancer mortality and the most common cancer in women [1]. Triple-negative breast cancer (TNBC) refers to breast cancer that tests negative for all of the following receptors: estrogen receptor (ER), progesterone receptor (PR), and human epidermal growth factor receptor 2 (HER2). As a consequence, TNBC can't be treated using hormone therapy [2]. The prognosis of patients diagnosed with TNBC is considerably worse compared to those diagnosed with non-TNBC. TNBC is more associated with distant relapse and affects the younger population to a larger extent [2].

Microscopic images of tissues are widely used when identifying and diagnosing the prognosis of cancer [3]. Most analysis today is done manually by a clinician, but these resources could be freed using machine learning [3]. A biomarker of interest, that can be studied using these images, is the amount of tumor-infiltrating lymphocytes (TILs) [4]. TILs are immune cells that have migrated from the bloodstream into the tumor. These are believed to carry prognostic information [4].

The project aims to use machine learning to extract high-level features, such as the number of tumor and immune cells, from digital scans of hematoxylin and eosin (H&E) stained tissue images, and in turn use these features to predict the patient's outcome, in a manner which lets us understand what the prediction is based on.

2 Theory

This section aims to explain some concepts that are important for this project. Section 2.1 contains a brief description of the inner workings of the computing systems known as artificial neural networks. Section 2.2 continues to describe convolutional neural networks, networks tailored to image processing. The task known as image segmentation is described in section 2.3. All these sections are related to machine learning. The next section, 2.4, aims to give a brief introduction to tumor-infiltrating lymphocytes, which are central to this project.

Any of these sections can be skipped if the reader is already familiar with the topic.

2.1 Artificial neural networks

Artificial neural networks (ANNs) are computing systems that are commonly used in machine learning applications. They were first demonstrated back in the 1961s, but due to the lack of training data and computing power they first become practical in the 2010s [5].

The structure of an ANN is inspired by biological neural networks, and they, therefore, share many characteristics. Both an ANN and a biological neural network consist of interlinked neurons, which fire if they receive enough signals from neighboring neurons. The neurons are ordered in a hierarchical structure which enables the network to process information on different levels of abstractions. A simple ANN can be seen in Figure 1. Each neuron has parameters called weights and biases. There is one bias per neuron and one weight for every incoming connection. The activation of a neuron is determined by the neuron's activation function. The activation function takes as input a weighted sum of the neuron's input plus the neuron's bias, formally

$$f(b + \sum_{i=1}^{n} x_i w_i), \tag{1}$$

where b is the neuron's bias x_i is the input i and w_i is the corresponding weight. There's a number of different activation functions that are commonly used. One example is the rectifier, which is defined as $f(x) = max(0, x)$, often known as ReLU [6]. It is crucial that the activation function is non-linear since it can be shown that multi-layered ANNs with linear activation functions can be rewritten as a single-layered neural network, defeating the purpose of having many layers.

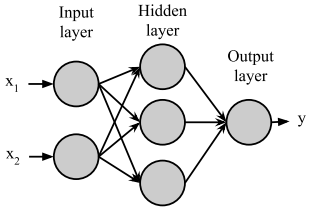


Figure 1: A simple artificial neural network with one input layer, one hidden layer and one output layer. Each neuron has one weight per input and one bias. In this network there are 17 trainable parameters.

The weights and biases of the neurons are the parameters that stores 'knowledge' in the network. They are not set manually, instead, their values are the result of the training of the network. The network can be trained using supervised learning. During training, input-output pairs are fed to the network. Every iteration, the network's prediction of the output is compared to the correct answer. A suitable error function is used to calculate the error, i.e. how wrong the network's prediction was compared to the ground truth. The derivative of the error with respect to every parameter in the network is calculated and the parameters are updated to decrease the error in the next iteration. This process is called backpropagation [7]. A network trained in this manner may be able to make valid predictions on samples previously unseen by the network, as long as they are from the same domain as the training samples.

2.2 Convolutional neural networks

All neurons in the hidden layer in the network in Figure 1 are connected to all neurons in the previous layer. Layers like this are referred to as *dense*. The problem with dense layers is that the number of trainable parameters quickly becomes too many to be practical for large networks. For example, if one would like to have an image with the resolution 256 x 256 as an input, it would require $256 \times 256 \times 3 = 196608$ input neurons (because each pixel consists of one value for every channel, red, green and blue). It is not hard to see how this would quickly get out of hand if all neurons in the next layer would have to be connected to each previous neuron.

This problem is solved with the invention of convolutional neural networks (CNNs) [8]. In this context, the word 'convolution ' is referred to the mathematical operation, not to something being complex, coiled, or twisted.

2.2.1 Kernel convolutions

CNNs use processes called kernel convolutions. An example of a result from a kernel convolution can be seen in Figure 2. During the process, a small matrix, the kernel, is sliding over the image, from the top left corner, row by row, to the bottom right. At every position the kernel will cover a set number of pixels, in this case 9 per channel since the kernel is 3 x 3. The covered pixels will in each step be multiplied with the corresponding values in the kernel, the weights. The sum of these products will then be the value of the corresponding pixel in the output image. The transformation of the input image depends on the weights in the kernel. In this case, all weights are 1/9 resulting in a blurred image.

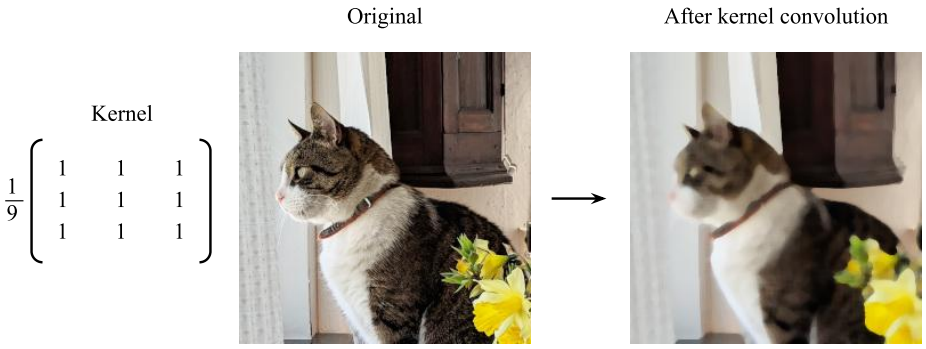


Figure 2: In a kernel convolution a kernel is sliding over the image. In the case of a 3 x 3 kernel, the kernel will cover 9 pixels, in each channel in the image, at each position. The covered pixels are multiplied by the corresponding values in the kernel. The sum of the products is the value for a new pixel in the output image. Note that this is done for every channel in the image.

2.2.2 Convolution layers

A convolution layer consists of one or more kernels each outputting an image. The resulting image is known as a *feature map*. The weights in the kernels will be a result of training just as in the regular ANN. Some kernels might be used by the network to detect edges and corners, but it's not possible to know for sure. One can only guess what feature will be useful for a specific task.

To save memory a convolutional layer is often followed by a *pooling layer*, which job is to downsample the feature map. See Figure 3 for an example of how the dimensions of an image or a feature map can change as a consequence of being processed by a convolution layer and a pooling layer. There are typically 3 channels in an image, red green, and blue. A standard 2D convolution layer processes all channel at the same time using a 3 x 3 x 3 kernel. After being processed by a convolution layer with 6 kernels the depth of the resulting feature map will now be 6. To save memory the feature map is downsampled using a pooling layer.

There are many types of pooling-layers. A common type is *max-pooling*, where a window is sliding over the image. In each position, the max value within the window is selected to be the output image value. A max-pool layer with a 2 x 2 window with a stride of 2 will result

in a feature map the quarter of the size of the input.

Sometimes there's a need for up-scaling a feature map. The easy way would be to just apply regular image up-scaling. A better way would be to use a convolution layer called *transposed convolution* [9]. Here the feature map is transformed by matrix multiplication. Once again the weights in the matrix are multiplied with the feature map is learned similar to the weights in the kernel.

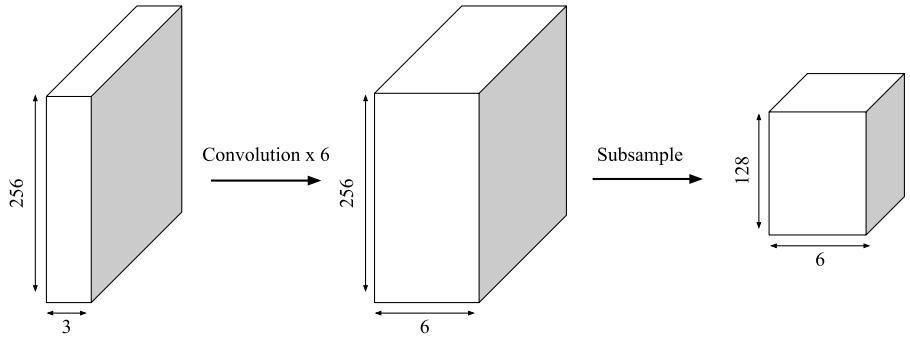


Figure 3: An example of how the dimensions of an image or feature map change after being processed by a convolution layer and a pooling (subsample) layer

A CNN is often made up of many successions of convolution layers and pooling layers. In 2012 Alex Krizhevsky won the ImageNet Large Scale Visual Recognition Challenge with Alexnet, consisting of five convolution layers and three pooling layers [10]. The advent of Alexnet is often seen as the start of the deep learning revolution [5].

2.3 Image segmentation

ANNs in general and CNNs, in particular, are useful for image processing tasks. Image classification is the use-case that most often comes to mind when discussing CNNs. The goal of this task is to classify images into predefined classes. The goal of image segmentation, however, is to determine what class each pixel in an image belongs to. The output of the network can be interpreted as a mask and is often referred to as a segmentation map, see Figure 4. The task is called *instance segmentation* if the goal is to also determine which *object* within one class a pixel belongs to. In applications where objects don't overlap, such as in nuclei segmentation, there is no practical difference between semantic segmentation and instance segmentation, since it's trivial to separate objects that don't overlap.

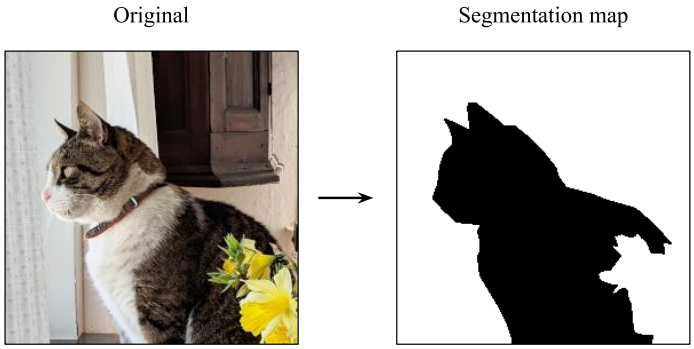


Figure 4: An example of semantic segmentation. In the image on the right hand side, all pixels are classified as pixels belonging to a cat or not belonging to a cat.

2.3.1 U-Net

U-Net is a convolutional neural network, designed for semantic segmentation, which was first proposed in 2015 [11]. It has since been used widely and the original paper is cited over 19 000 times. A schematic of the U-Net architecture can be seen in 5. The architecture is shaped like the letter U, hence the name. It has one contracting part and one expanding part. The contracting part consists of a series of convolutions layers with max-pooling layers intertwined. The expanding part alone is almost identical to a CNN that might be used in a classification task. However, since the goal is segmentation, spatial information needs to be retained. That is what the expanding part is for. The expanding part consists of a series of convolutional layers, this time with transposed convolution layers in between. For every level in the U-Net, the information in the contracting part is concatenated with the expanding part. This retains the spatial information, that would otherwise be lost.

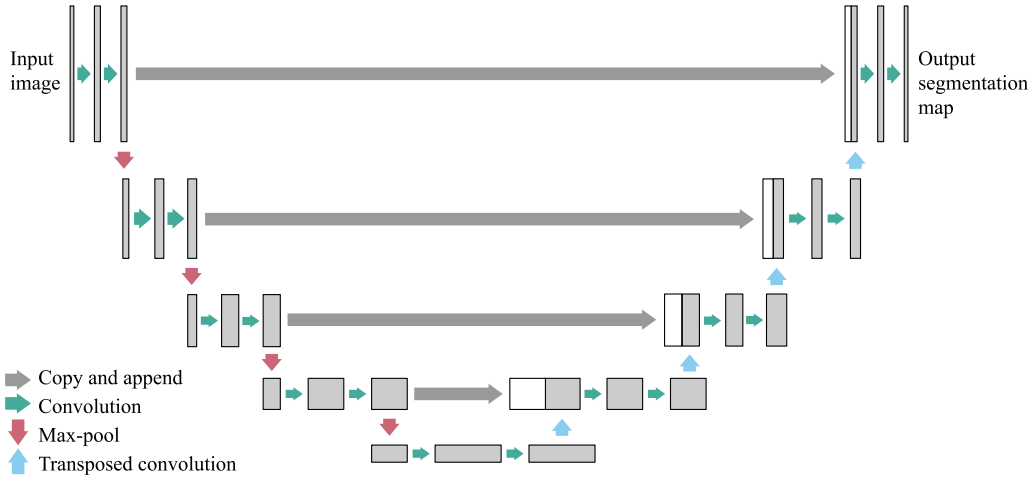


Figure 5: The U-Net architecture. The arrows denote the operation at each step. The relative height of the rectangles represent the resolution of the feature maps, while width represents the depth.

2.4 Tumor-infiltrating lymphocytes

Tumor-infiltrating lymphocytes (TILs) are immune cells that have migrated from the bloodstream into a tumor as a part of the antitumor immune response. In some breast cancers, like TNBC, a large amount of TILs has been linked to a positive outcome for the patient [4].

TILs can be seen in tissue images stained with H&E. In 2015 the International TILs Working Group released a recommendation on how to estimate the level of TILs in tissue images as a way to promote an international standard [4]. TILs can be divided into stromal TILs and intratumoral TILs as shown in Figure 6. Stromal TILs are located in the tissue surrounding the tumor cells, whereas intratumoral are located among the tumor cells. The score of TILs is defined by the International TILs Working Group as the percent of the stromal area covered by TILs [4].

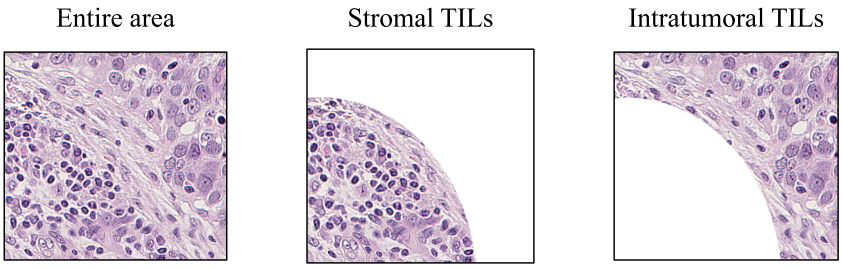


Figure 6: The different regions relevant to the scoring of TILs. The dark cells are immune cells (TILs in this context) while the brighter and larger cells are tumor cells.

Intratumoral TILs are believed to be of importance since they have direct interaction with the tumor cells. However, the TILs Working Group does not recommend using these as a metric since the scoring of such TILs is harder to reproduce. They instead suggest only to focus on the area covered with stromal TILs.

3 Project overview

The aim is to find new biomarkers in tissue images for assessing the outcome of patients diagnosed with TNBC. New biomarkers could be used to aid clinical decision making, both manually or by using models. High-level features, such as the number of immune cells and their locations, are extracted from tissue images using machine learning. The prognostic power of these featured are then evaluated by measuring how well they can predict an outcome. An overview of this procedure can be seen in Figure 7.

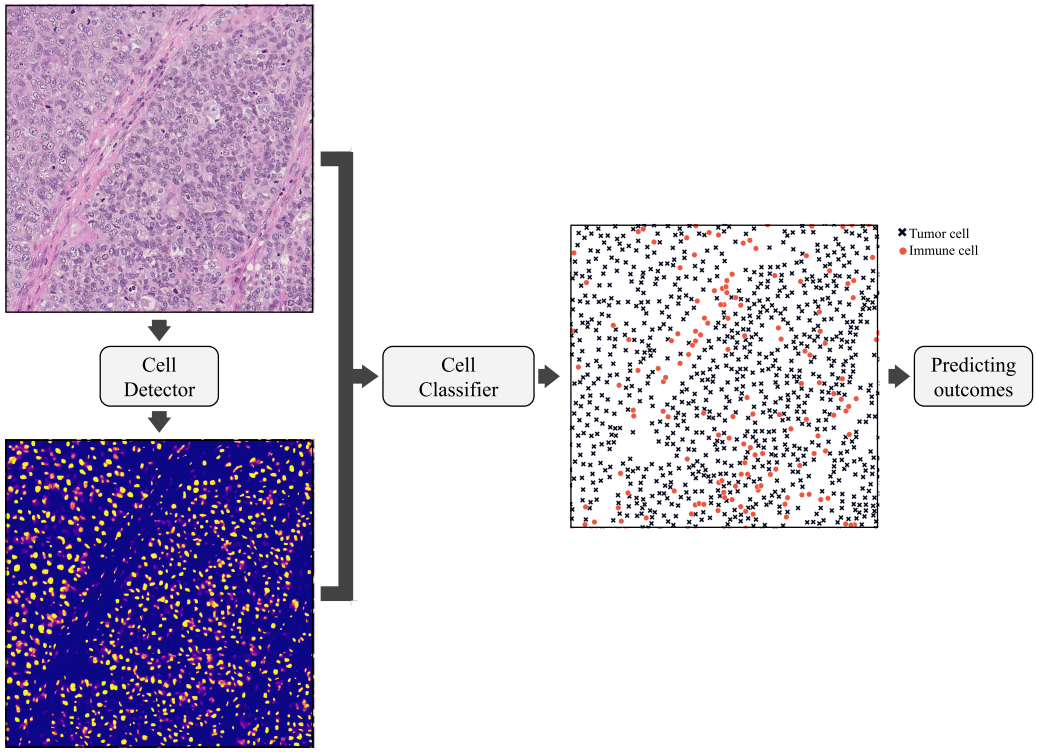


Figure 7: A U-Net-based cell detector and a cell classifier are used to extract high-level feature, such as the number of immune cells and tumor cells, in images of cancerous tissue. These features are in turn used to predict patient outcomes.

Two machine learning models are used for feature extraction. The first one, the *cell detector*, is used for detecting cells in tissue images. This model is a U-Net trained on a large set of public data. The output from this model is a segmentation map showing the location and mask of every cell. The second model, the *cell classifier*, is a logistic regression model that used an image and its mask to classify every detected cell as tumor cell or immune cell. The cell classifier was trained on a small number of manually annotated samples.

The amount of tumor cells and immune cells and their spatial distribution can then be extracted. These are the main features used in the search for bio-markers in a cohort of 155 TNBC patients.

4 Data

Four different datasets are used in the project, see table 1. This section aims to describe the characteristics of these datasets and what they are for. All datasets consist of H&E-stained tissue images. The datasets are called Quip, Bns, MoNuSeg, and TNBC-cohort.

Quip, Bns, and MoNuSeg are public and fully annotated, meaning that a mask exists for every cell in every image. These sets have been used for training and evaluating the cell detector. The TNBC-cohort-datasets were not fully annotated. These sets consist of tissue images from the triple-negative cohort that was the main interest for this project.

Name	**Resolution**	**Size**	**Zoom**	**Annotated**	**Cancer**
Quip	4000^2	3600	x40	Yes, binary	Breast
MoNuSeg	1000^2	44	x40	Yes, binary	Multi-organ
Bns	512^2	33	-	Yes, binary	TNBC
TNBC-cohort (TMA)	3000^2	542	20x	Partially, multi-class	TNBC
TNBC-cohort (WSI)	800000^2	221	20x	No	TNBC

Table 1: Summary of the datasets used. All datasets consists of H&E-stained tissue images.

4.1 Quip

This dataset is used to train the U-Net based cell detector. The dataset, published in 2020, has billions of annotated cells which by far makes it the largest of its kind [12]. It consists of tissue images of several types of cancer, but only the images of breast cancer are used in this project. The dataset was automatically annotated by a U-Net trained on synthetic data generated by a neural network. The dataset has been manually quality controlled by the authors. Each image has an annotation quality rating. Only the samples that received high ratings are used in this project. The images are captured at a magnification of x40.

4.2 MoNuSeg

MoNuSeg was published in 2018 as a part of the 2018 Multi-Organ Segmentation Challenge [13]. As the name suggests, it consists of images from a diverse set of organs and diseases. The dataset consists of 44 images with the resolution 1000 x 1000, captured at x40. This dataset is used to evaluate the performance of the U-Net trained on Quip. Since the results from the competition are published, we can compare the performance of our model to the ones on the competition leader board [14].

4.3 Bns

Bns was published in 2017 [15]. It consists of 33 images á 512 x 512 pixels of triple-negative breast cancer tissue. Despite being a small dataset it's of interest since all patients were triple-negative. The dataset is therefore closer to the target domain. Because of this Bns is used to evaluate the performance of the U-Net model trained on Quip.

4.4 TNBC-cohort

These images are used to find biomarkers for TNBC outcome. The dataset contains tissue-images captured at x20 magnification. There are 221 patients in this cohort, all of them were diagnosed with triple-negative breast cancer. Of the 221 patients, 155 received treatment, and 66 did not, 171 survived and 50 did not.

The images are scans of tumors that have been surgically removed from the patients as a part of the treatment. The images exist in two forms, whole slide images (WSIs) and tissue array images (TMAs), see Figure 8. The WSIs are very large and therefore hard to work with and study. The idea of TMAs is to capture a region of the WSI that gives a fair representation of the whole slide while being easier to study thanks to the smaller form factor. When making TMAs, a hollow needle is used to extract tissue cores from a tumor. Cores from multiple patients are then fixed in an array where they can be sliced into thin circular sheets. This allows for preparing and scanning tissues from multiple patients in parallel [16].

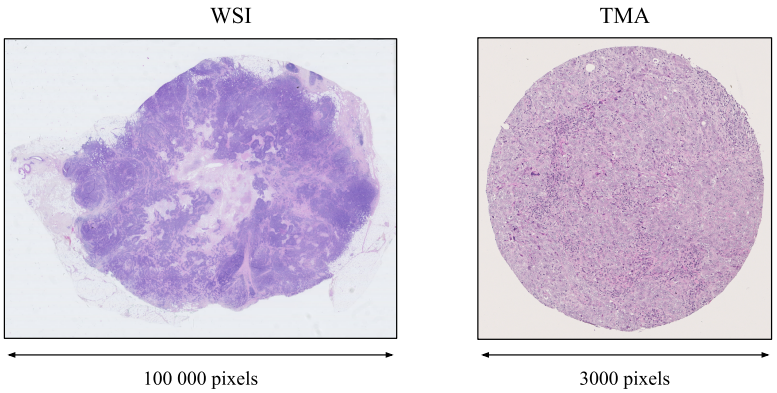


Figure 8: An example of a whole-slide image (WSI) and a tissue array image (TMA). Note the difference in resolution.

A subset of the TMAs is partially annotated. The boundary and the class of 658 tumor cells and 593 immune cells are labeled. These labels are used to train and evaluate the cell classifier that is used to classify cells identified by the cell detector.

5 Methodology

This section aims to describe the implementation and the results of each task. The cell detection is described in 5.1. The cell classification is described in 5.1.The features extraction using the trained models is described in section 5.3 and the prediction of outcomes is described in section 5.4.

5.1 Detection of cells

The cell detection model is a U-Net trained on the Quip dataset. The U-Net architecture is used for semantic segmentation, but since the cells don't overlap, cell instances can easily be

inferred from a segmentation map. The U-Net in this project differs in some ways from the architecture described in the original U-Net paper. In the original, the output is a tile with two channels, one for cells and one for background whereas my model uses one additional channel for the cell boundaries [11]. This should improve the separation of neighboring cells as shown by Naylor et al. (2019) [17]. The boundary masks are made from the original mask by eroding the mask by 2 pixels and subtracting it from the original.

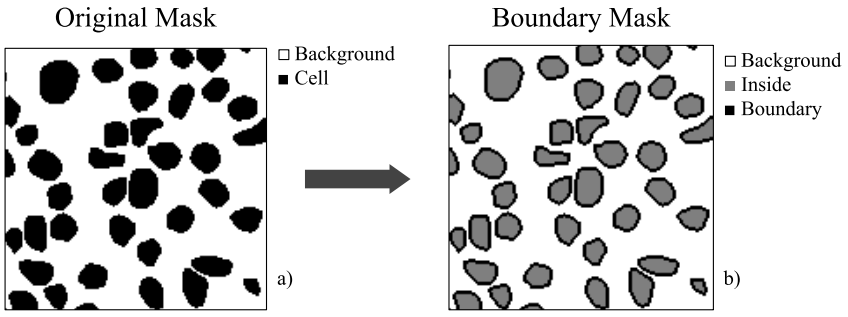


Figure 9: The original mask a) has two channels, one for background and one for cells. The boundary mask b) is made from the original mask and has one channel for background, one for cell insides and one for cell boundaries

To further improve the separation of neighboring cells a weighted cross-entropy loss function is used during training, just like in the original U-Net paper [11]. Weight maps are used in the loss function to penalize miss-classified pixels between neighboring cells. The weight maps are calculated based on the pre-existing segmentation map. The values of the weight map are based on the distances between neighboring cells. The weight w for a position $\mathbf{x}$ in the weight map can be calculated by the following

$$w(\mathbf{x}) = w_c(\mathbf{x}) + w_0 \cdot \exp\left(-\frac{(d_1(\mathbf{x}) + d_2(\mathbf{x}))^2}{2\sigma^2}\right), \tag{2}$$

where $w_c(\mathbf{x})$ are class weights to make up for the imbalance of the class frequencies, these are however set to $w_c(\mathbf{x}) = 1$ in this project. d_1 and d_2 denotes the distance to the border of the nearest and second nearest cell. The constants w_0 and σ are set to 10 and 5. A weight map example can be seen below.

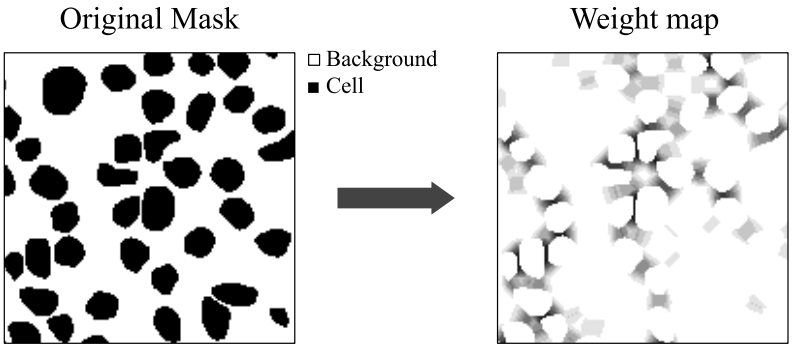


Figure 10: The weight map is defined by equation 2 based on the locations of the cells in the original mask. The weight map is used in the loss function to encourage the the separation of neighboring cells.

Usually, the weight maps are computed before training, but here the weight maps are generated during training. This slows down the training process but allows for more rapid development since there is no overhead in changing how the weight maps are generated.

The network uses a categorical soft-max function for its output activation. The activation function is defined as

$$p_k(\mathbf{x}) = \frac{\exp(a_k(\mathbf{x}))}{\sum_{k'=1}^{K} \exp(a_{k'}(\mathbf{x}))}, \tag{3}$$

where $a_k(\mathbf{x})$ is the activation for channel k at location $\mathbf{x}$ in the feature map. K is the number of channels, three in this case. This makes $p_k(\mathbf{x}) \approx 1$ for the channel k that has the strongest activation. The weighted categorical cross entropy loss is the calculated by

$$E = \sum_{\mathbf{x}} w(\mathbf{x}) \log(p_{l(\mathbf{x})}(\mathbf{x})), \tag{4}$$

where $w(\mathbf{x})$ is the weight function from equation 2 and $p_{l(\mathbf{x})}(\mathbf{x})$ is deviation from 1 for the true label for position $\mathbf{x}$. Thus, the loss is penalized for wrongly classified pixels with extra emphasis on pixels between neighboring cells. This lowers the risk of detecting one large cell where there are really two cells close to each other.

The model takes as input a boundary map, a weight map, and an image tile with dimensions of 512 x 512 x 3, where 3 refers to the channels red, green, and blue. During training images from the training set are selected at random, one per epoch. The image is split into 16 pieces and one tile is selected at random from each piece, see Figure 11. To decrease the number of tiles with few or no cells, only training images with 2000 or more cells are used. Image augmentation is not used since the training set is large and it is unlikely that the same tile will be selected more than once during training.

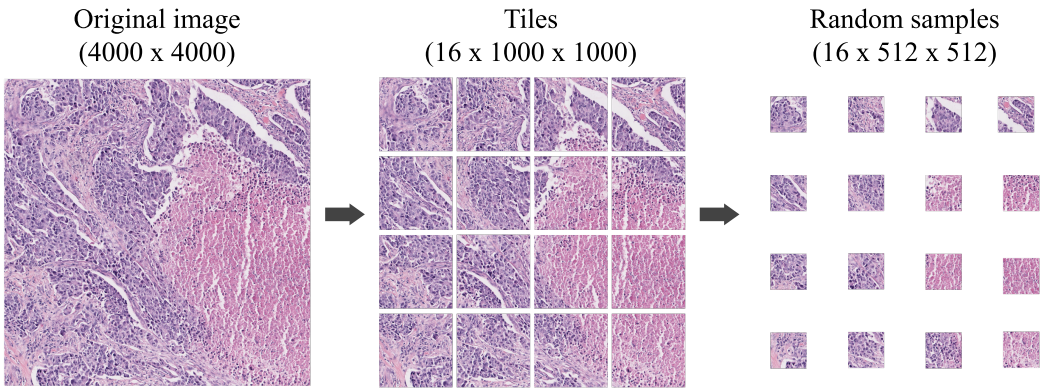


Figure 11: During training, images from the Quip dataset are picket at random. These are split into 16 tiles from which one 512 x 512 tile is sampled at random. This is done once per epoch.

The model was trained on a Nvidia GeForce GTX 1080 Ti GPU. It was trained for 1000 epochs and was finalized after 6 hours. Adam was selected as as optimizer with the learning rate set to 1e-5.

During the prediction of an image larger than 512 x 512, a sliding window with stride 256 is used. The predictions are then assembled and overlapping predictions are averaged. Only the inside-channel (see Figure 9a) is used in the final segmentation map. The cutoff is set to 0.2. A dilation of 2 pixels is then applied to make up for the pixels lost when creating the boundary mask. As a final step, all cells smaller than 5 pixels are removed, as they are deemed to be false positives. The choice of 5 pixels as size limit is somewhat arbitrary. (The smallest among the annotated cells in the TNBC-cohort is 29 pixels in size.)

The cell detector is evaluated on the MoNuSeg-set, the Bns-set, and images from the Quip-set that were not used for training. Intersection over Union (IoU) is used as metric. The results can be seen in Table 2, accompanied by the highest scores from the MoNuSeg challenge and from the authors of the Bns-dataset. With a ÌoU of 0.66, the cell detector would have ranked 7th place in the MonNuSeg challenge.[14].

	IoU by model		
Dataset	**Cell detector**	**DeConvNet** [15]	**CHUK & IMSIGHT** [14]
Bns	0.44	0.81	-
MoNuSeg	0.66	-	0.69
Quip	0.70	-	-

Table 2: Performance of the cell detector on different datasets compared to other models.

5.2 Classification of immune cells and tumor cells

A classifier is used to separate immune cells from tumor cells among the detected cells in the tissue images from the TNBC-cohort. There are several ways in which tumor cells and immune cells differ in appearance. Examples of tumor cells and immune cells can be seen in Figure 12.

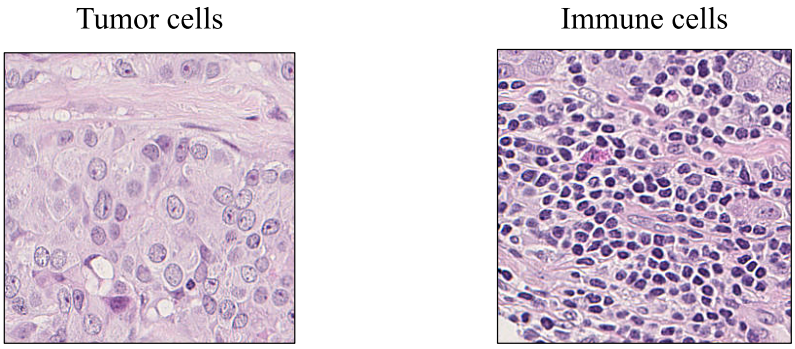


Figure 12: Examples of tumor cells and immune cells.

The characteristics can be further revealed by studying the annotations from the TNBC-cohort. The annotations reveal that tumor cells are in general larger, brighter, and, more heterogeneous than immune cells, as shown Figure 13. The brightness is measured by the average intensity of the pixels that cover the cell whereas the heterogeneity is represented by the intensity variance. Note that word *intensity* have the opposite meaning in the fields of image processing and histology. In image processing, high intensity usually refers to high pixel values, which makes the pixels look bright. In histology a cell with high intensity is often a cell that has absorbed a lot of dye, making it darker. In this document *intensity* is synonymous with brightness. The area is simply the sum of pixels covered by the cell.

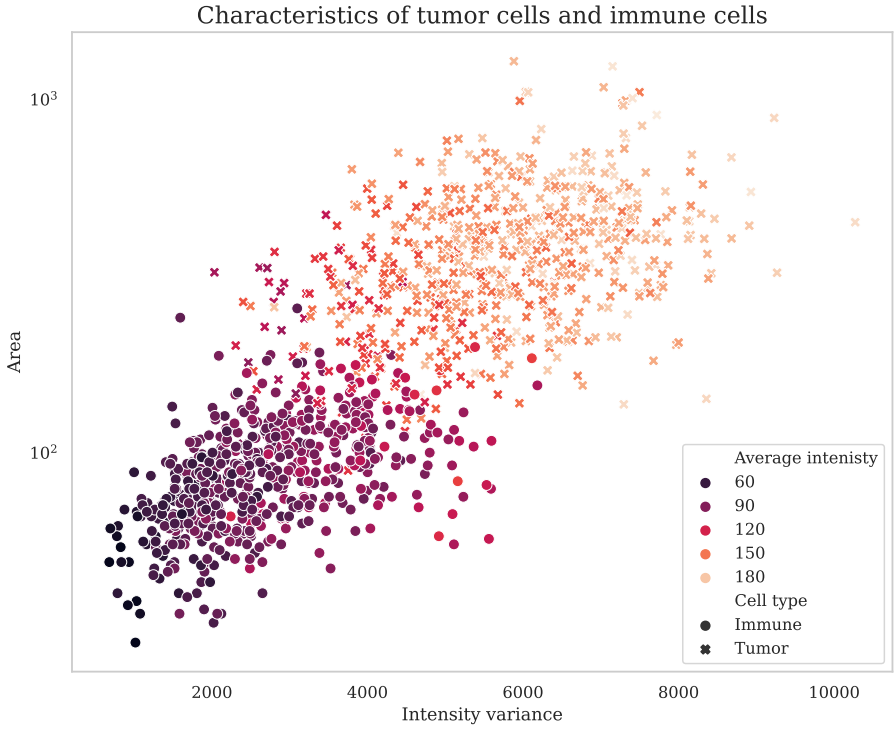


Figure 13: Based on the annotations of the images from the TNBC-cohort tumor cells are larger, brighter and more heterogeneous than immune cells. The intensity (brightness) of a cell is denoted by the color. The cell type is denoted by shape.

The cell classifier is a logistic regression model using intensity, intensity variance, and area as features. The model was implemented using scikit-learn [18]. Five-fold cross-validation was used on 80 % of the samples. The model was then trained on the whole validation set and evaluated on the remaining 20 %. The model was also evaluated on the cells that were detected by the cell detector. This indicates the importance of having a perfect segmentation mask since a detected cell might contain some background pixels or just a piece of the cell. Among the cells in the test set, 4 immune cells were not detected at all by the cell detector. The 4 cells not found by the cell detector were removed from the test-set to evaluate the classifier on the cells that were found. The classifier was evaluated on the remaining 253 cells detected by the cell detector.

The ROC curves and AUC-scores for the classification of cells in the validation set, test set, and detected cells can be seen in Figure 14. Note that the test AUC is higher than the validation AUC since more samples were used. A possible explanation for the perfect AUC-score is that the entire dataset might only contain cells that are easy to distinguish.

The model is trained on all samples. Note that immune cells and tumor cells are not the only cell types found in the tissue images. These cells will be miss-classified using this method.

The other types of cells do seem to be a lot fewer in number, compared to immune and tumor cells.

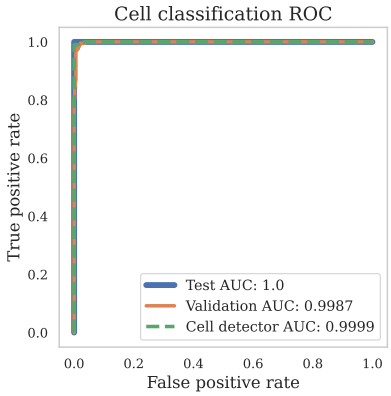


Figure 14: The evaluation of the cell classifier using AUC as metric.

5.3 Extracting features from model outputs

The cell detector described and the cell classifier described in section 5.1 and 5.2 can be used to detect and classify cells in a tissue image of arbitrary size. The detection and classification are carried out for the TMAs and WSIs alike. However, detecting all cells in a single WSI could take well over a day with available equipment, even when the white backgrounds are excluded.

The large images are sampled in order to capture the overview offered by the WSIs while still finishing the computation within an acceptable time frame, see Figure 15. A large part of the WSI consists of a white background. There are also dark patches that are not a part of the tissue. The background and non-tissue patches are excluded using a mask defined by $M(\mathbf{x}) = 0.1 < I(\mathbf{x}) < 0.9$, where $I(\mathbf{x})$ is the average intensity at pixel $\mathbf{x}$. As a final step, all areas of the mask with an area less than 10 % of the whole are removed. A number of 1024 x 1024 patches are then sampled from tissue covered by the mask.

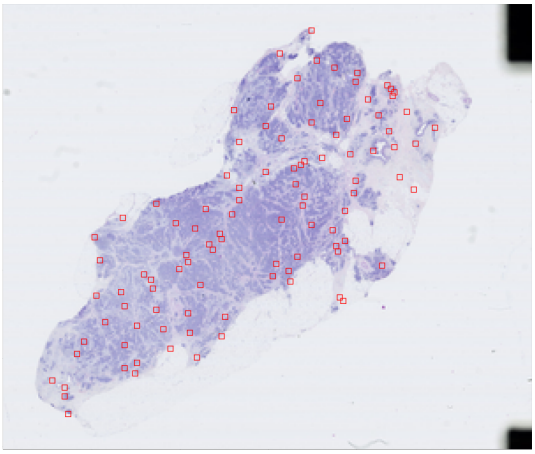


Figure 15: The WSI is sampled to reduce computation time. The samples are only picked within a mask to avoid sampling the background or the dark patches in the upper and lower right corner. Each square represents a 1024 x 1024 patch.

One hypothesis is that the spatial distribution of cells carry information about the likelihood of recurrence in TNBC. To aid the processing of cell locations the cells are reduced to points located roughly in the center of each cell. A number of spatial features can then be extracted using single point locations.

The first of these features is a measurement of how unevenly the cells are spread. It's based on the Ripley's K [19] metric and is defined as

$$K(r) = \frac{1}{n}\sqrt{\sum_{i \neq j} I(d_{ij} < r)}, \tag{5}$$

where n is the number of cells, d_{ij} is the distance between cell j and i, $I = 1$ if the distance $d_{ij} < r$, otherwise $I = 0$. This results in a large $K(t)$ if the cells are clustered in groups rather than spread out evenly. The search distance r is a parameter that has to be set manually. The metric is computed for tumor cells and immune cells separately.

The second feature is inspired by the metric discussed in section 2.4. The aim is to divide the immune cells into stromal TILs and tumor TILs. This is done by counting the number of immune cells and tumor cells within a radius r of an immune cell. If the immune cell has more than twice the number of immune cells compared to the number of tumor cells within its proximity it is classed as a stromal TIL. If it's the other way around, having twice as many tumor cells in its proximity, it is classed as a tumor TIL. The number of stromal TILs $N_{it}(r)$ and tumor TILs $N_{is}(r)$ can be expressed like this,

$$\begin{cases} N_{it}(r) = \sum_{i=1}^{N_i} I\left(\sum_{j \neq i} I(d_{ij} < r) > 2 \sum_{t=1}^{N_t} I(d_{it} < r)\right) \\ N_{is}(r) = \sum_{i=1}^{N_i} I\left(\sum_{j \neq i} I(d_{ij} < r) 2 < \sum_{t=1}^{N_t} I(d_{it} < r)\right), \end{cases} \tag{6}$$

where N_i and N_t are the total number of immune cells and tumor cells, d_{ij} is the distance between immune cell i and j, d_{it} is the distance between immune cell i, and tumor cell t

and $I(s) = 1$ if the statement s is true.

Each of these features is computed for all images. There are two TMAs per patient and an arbitrary number of WSI patches. An average of each feature is used to deal with multiple images. The features are then combined with the manually annotated STR-estimates, discussed in section 2.4, and the corresponding patient data. This includes the patient's age at diagnosis, the size of the tumor, and the number of positive lymph nodes detected at surgery. All features are found in Table 3.

Metric	**Feature set**	**Description**
A_i	Computed feature	Area in pixels covered by immune cells
A_t	Computed feature	Area in pixels covered by tumor cells
N_i	Computed feature	Total number of immune cells
N_t	Computed feature	Total number of tumor cells
$K_i(r)$	Computed feature	Spacial heterogeneity of immune cells
$K_t(r)$	Computed feature	Spacial heterogeneity of tumor cells
$N_{it}(r)$	Computed feature	Number of stromal TILs
$N_{is}(r)$	Computed feature	Number of tumoral TILs
Age	Patient data	The patient's age at diagnosis
Size	Patient data	The tumor size in millimeters
N_n	Patient data	The number of positive lymph nodes at surgery
STR	Estimated stromal TILs	The Stromal TILs Ratio estimated by using the method proposed by the International TILs Working Group. This metric is explained in section 2.4.

Table 3: The features used for predicting outcomes of TNBC.

5.4 Predicting TNBC outcomes

Logistic regression models are used to investigate how well the features in Table 3 can predict an outcome for a patient. The outcomes investigated are distant relapse, local relapse, and all-cause mortality. Note that these are predicted independently of each other. The outcome is only predicted for the 155 patients who received adjuvant treatment. The variance is much higher in the group of patients who didn't receive adjutant treatment, which makes it hard to compare them to the ones who received adjutant treatment.

Nested-five fold cross-validation is used to avoid the risk of overfitting. Combinations of features are selected based on their AUC-scores on the validation sets. The select features are then evaluated on the test set, which contains samples not included in the train and validation sets. All features are normalized, before model fitting and prediction, by subtracting the mean and dividing by the variance.

The features can be divided into the following sets: computed features, estimated stromal

TILs, and patient data. In order to compare the predictive power of each group, models are fitted to features constrained to each set, but also to all features in all sets. The selected features and their validation-score can be seen in Table 4. Note that sometimes features from only one subset of features resulted in the best validation score.

The five-fold cross-validation results in five models for every combination of features, for every outcome, and for both image types. The AUC scores presented are the total test scores for the prediction made by all five models fitted to a specific set of features. Hence, the scores therefore measure the set of features' ability to predict an outcome, and not the performance of a specific model instance.

The models are fitted using the Python library statsmodels [20]. The number of WSI patches used is 800.

	Distant relapse		**Local relapse**		**Mortality**	
Feature set	**Selected Features**	**AUC**	**Selected features**	**AUC**	**Selected features**	**AUC**
All features (WSI)	$N_{is}(100)$, $K_i(100)$	0.71	N_i, $N_{it}(100)$, $N_{is}(100)$	0.73	Age, Size, $K_t(100)$	0.73
Computed metrics (WSI)	$N_{is}(100)$, $K_i(100)$	0.71	N_i, $N_{is}(100)$, $N_{it}(100)$	0.73	N_i, $N_{it}(100)$, N_t	0.70
Estimated stromal TILs (WSI)	STR	0.61	STR	0.69	STR	0.62
All features (TMA)	STR, A_i, $K_i(100)$, N_n, N_i	0.69	STR	0.54	STR, Size, A_i, Age N_t, N_i	0.68
Computed metrics (TMA)	A_i, $K_i(100)$, N_t, N_i	0.68	A_i, $N_{it}(100)$, N_t	0.52	N_t, $N_{is}(100)$, $N_{it}(100)$	0.61
Estimated stromal TILs (TMA)	STR	0.49	STR	0.54	STR	0.45
Patient data	N_n	0.56	Size	0.44	Age	0.65

Table 4: The features are selected based on their validation score. These are the features that scored highest on the validation sets. The features are picked from different feature sets. Note that in some cases additional sets of features did not yield a higher validation score.

The selected features' ability to predict the respective outcome is then evaluated using the test sets. New models are fitted to both the training set and the validation set. The fitted models are then used to predict the samples in the test tests. The resulting ROC-curves for the prediction of distant relapse, local relapse and mortality can be found in Figure 17, 18, and 19.

Each features' ability to predict outcomes without combinations is also evaluated. The features are grouped by source. The features based on the WSIs and TMAs can be found in Table 6 and 5, whereas the features based on patient data can be found in Table 7. These tables also include the average coefficient from the logistic regression models. This gives

an indication whether a feature is associated with the increased or decreased risk of distant relapse, local relapse or mortality.

The utility of increasing the number of WSI-samples is investigated by measuring the AUC using different number of WSI-patches. The results can be seen in Figure 16. Note that x-axis is logarithmic and that the utility of more samples quickly diminishes, but even more so for distant relapse and mortality. Also note that sometimes less than ten WSI-patches yields a higher AUC than when using hundreds of patches. This suggests that some regions of the tumor carry much more predictive information than others.

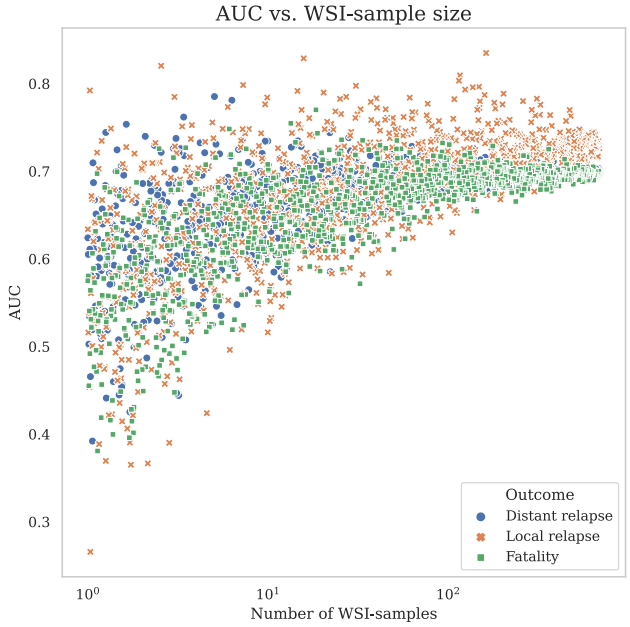
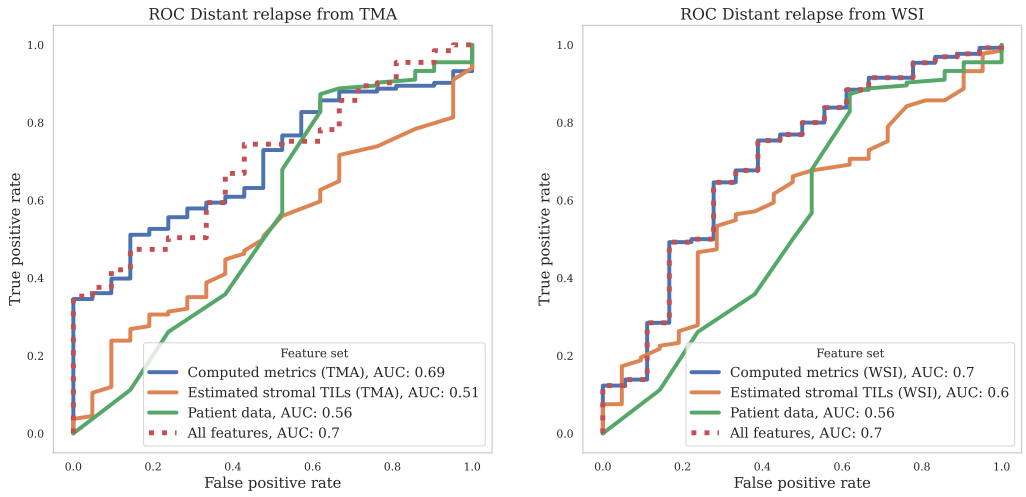
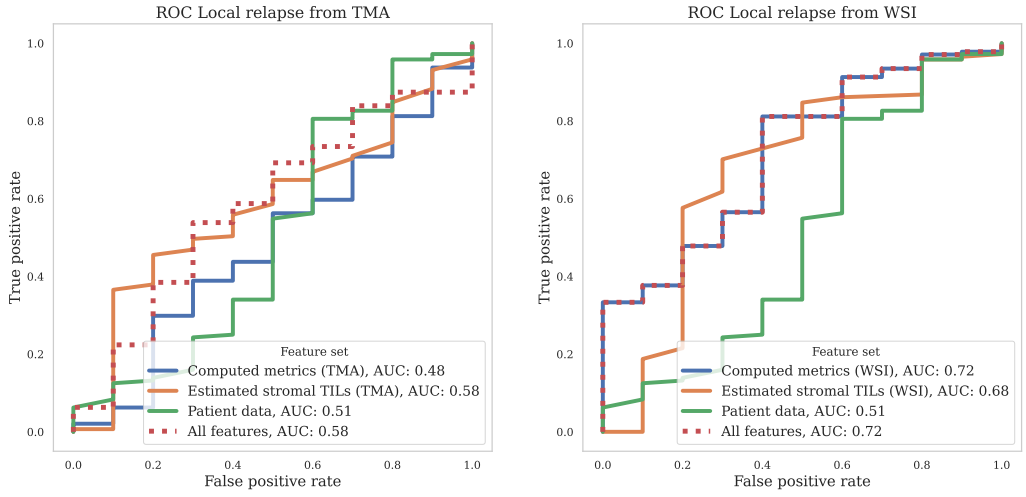


Figure 16: Prediction accuracy increases with more WSI-samples. But the returns quickly diminishes. Note that the x-scale is logarithmic.



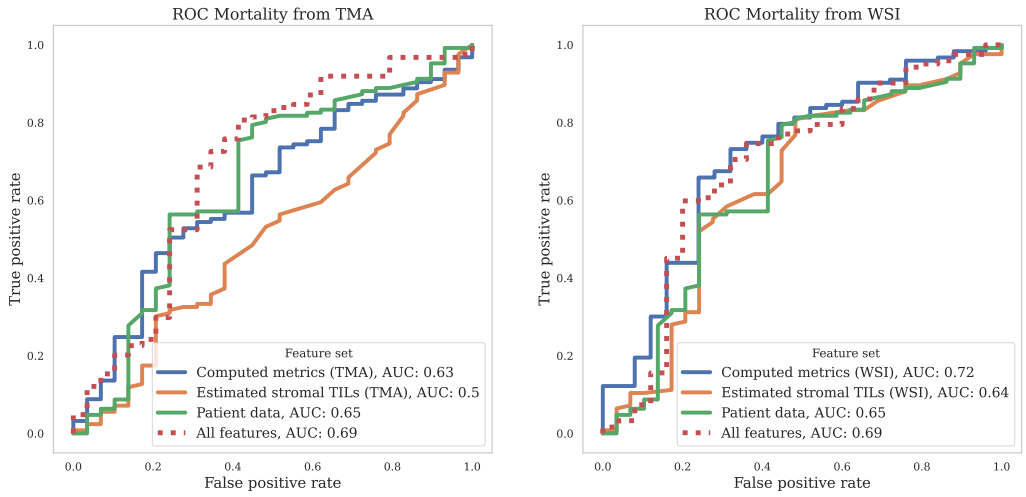
(a) (b)

Figure 17: a) The highest AUC, when predicted distant relapse using TMAs, is achieved using a combination of computed features combined with estimated stromal TILs (see Table 4). b) When using WSIs the models trained on the computed features perform better than others or combination of other feature sets.



(a) (b)

Figure 18: a) When using TMAs to predict local relapse, estimated stromal TILs is the only feature that might be used as a predictor but doing so results in a low AUC, fairly close to 0.5. b) Models fitted to computed WSI-features performs better than other feature sets, or combinations of feature sets. Patient data (patient age, size of tumor, and number of lymph nodes) doesn't seem to be relevant when predicting local relapse.



(a) (b)

Figure 19: Mortality is best predicted using a combination of features sets. Data from the WSIs seems achieve better predictions compared to the TMAs.

Distant relapse			**Local relapse**			**Mortality**		
Metric	**AUC**	**Coeff.**	**Metric**	**AUC**	**Coeff.**	**Metric**	**AUC**	**Coeff.**
$K_i(100)$	0.71	0.65	A_i	0.69	-1.87	A_i	0.70	-1.27
N_i	0.71	-1.49	STR	0.68	-1.25	N_i	0.69	-1.18
A_i	0.71	-1.56	N_i	0.67	-1.61	$K_i(100)$	0.67	0.53
N_t	0.65	-0.62	$K_i(100)$	0.64	0.58	$N_{is}(100)$	0.66	-1.78
$N_{it}(100)$	0.65	-0.69	$N_{is}(100)$	0.63	-2.89	STR	0.64	-0.55
A_t	0.65	-0.68	N_t	0.62	-0.70	A_t	0.63	-0.55
$N_{is}(100)$	0.64	-1.51	$N_{it}(100)$	0.61	-0.80	N_t	0.63	-0.52
$K_t(100)$	0.64	0.54	A_t	0.59	-0.74	$N_{it}(100)$	0.60	-0.39
STR	0.60	-0.56	$K_t(100)$	0.49	0.26	$K_t(100)$	0.59	0.53

Table 5: The different features' ability to predict outcomes from WSIs. The coefficients are the averages from the five models produced by the cross-validation. A positive coefficient indicates that a feature is positively correlated with the outcome.

Distant relapse			Local relapse			Mortality		
Metric	**AUC**	**Coeff.**	**Metric**	**AUC**	**Coeff.**	**Metric**	**AUC**	**Coeff.**
N_i	0.64	-0.99	STR	0.58	-0.49	N_i	0.60	-0.49
A_i	0.62	-0.71	$N_{is}(100)$	0.55	-0.56	$N_{is}(100)$	0.60	-0.62
$N_{is}(100)$	0.57	-1.45	$K_t(100)$	0.41	-0.21	A_i	0.58	-0.36
N_t	0.54	-0.24	$K_i(100)$	0.41	0.14	N_t	0.53	-0.24
STR	0.51	-0.31	N_i	0.39	-0.31	A_t	0.52	-0.18
$N_{it}(100)$	0.48	0.04	A_i	0.34	-0.18	STR	0.50	-0.10
A_t	0.42	-0.03	$N_{it}(100)$	0.34	0.12	$K_i(100)$	0.48	-0.09
$K_t(100)$	0.41	0.20	A_t	0.31	0.27	$K_t(100)$	0.45	0.08
$K_i(100)$	0.41	-0.21	N_t	0.30	-0.03	$N_{it}(100)$	0.45	-0.02

Table 6: The different features' ability to predict outcomes from TMAs. The coefficients are the averages from the five models produced by the cross-validation. A positive coefficient indicates that a feature is positively correlated with the outcome.

Distant relapse			Local relapse			Mortality		
Metric	**AUC**	**Coeff.**	**Metric**	**AUC**	**Coeff.**	**Metric**	**AUC**	**Coeff.**
N_n	0.56	0.33	Size	0.51	0.39	Age	0.65	0.58
Age	0.53	0.13	N_n	0.46	0.10	Size	0.58	0.33
Size	0.26	0.04	Age	0.36	0.12	N_n	0.53	0.33

Table 7: The patient data-points' ability to predict outcomes. The coefficients are the averages from the five models produced by the cross-validation. A positive coefficient indicates that a feature is positively correlated with the outcome.

6 Discussion and future work

In this section, I discuss results presented in section 5. I also bring up ideas on how to improve the feature extraction and how to further explore how the features are related to patient outcomes.

6.1 Cell detector and classifier

Looking at the evaluation results in Table 2, we can see that the model's performance on the Bns dataset is surprisingly bad, despite it only containing breast cancer tissue, just like the training data. Compared to the MoNuSeg set, Bns is, lesser-known and might be of lower quality. This is one possible explanation of the low IoU. The performance on the MoNuSeg set is on the contrary good, despite the model solely being trained on breast cancer tissue, and none of the MoNuSeg images.

It's difficult to evaluate how well the cell detector performs on the TNBC-cohort. A clue can be found by investigating the missing cells in the classification test set when evaluating the cell detector. Approximately 1.5 % of the cells were not registered by the cell detector. This gives an idea of the false negative rate, but we have no way of knowing the false positive rate. The model seems to me good enough, but could probably be improved by increasing the training duration, utilizing color- and/or stain-normalization, and using image augmentation during training.

The cell classifier seems almost to good to be true, and one should be skeptical of the high AUC-test-score. One way of further evaluate the classifier would be to take a random sample from the detected and classified cells and manually count the number of true and false positives. The results also seem overly optimistic when considering the other cell types. It might be trivial to separate immune cells from tumor cells, but it is likely much more difficult to separate, for example, 3, 4, or 5 different types of cells. Since we are only dividing the cells into tumor cells and immune cells, all cells that are of neither type will be miss-classified.

To streamline the segmentation-classification pipeline, one could use the classifier to classify binary annotations in the segmentation training data, and then train a multi-class segmentation model using the classified annotations. This would result in less overhead during detection and classification since it would all be done in one step.

6.2 Feature extraction and prediction of outcomes

To some degree it seems to be possible to predict distant relapse, local relapse, mortality, using our method. By looking at the ROC-curves in Figure 17-19 it can be seen that some groups of features are more important than others, depending on which outcome to predict. The exact features used can be seen in Table 4.

Figure 17 shows the ROC-curves for distant relapse predictions. When looking at the curves and the AUC-scores, we notice that there doesn't seem to be any significant improvements by using WSIs over TMAs. Furthermore, we can see that the computed features perform better than the estimated STR in both cases.

Figure 18 shows the ROC-curves for predicting local relapse. In contrary to the distant relapse prediction, using WSIs seem to increase the prediction accuracy. In fact, all computed TMA features, and patient data seem to result in predictions no better than a coin-flip.

Figure 19 shows the ROC-curves for predicting fatality. Similar to predicting local relapse, using data from WSIs seems to yield better predictions. Patient data seems to be more important for predicting this outcome, which is not surprising since age is highly correlated with overall fatality.

For all predictions, except for predicting local relapse using TMAs, using computed features yields equal or better results than the estimated TILs. This might not be a fair comparison since there are more computed features than estimated features, thus increasing the odds that the best predictor is a computed one. A logical next step would be to try interactions between well-performing terms to see whether the prediction could be improved.

We can study the sign of the coefficients in Table 5-7 to figure out if any single feature is associated with a good or bad outcome. For all three outcomes few immune cells N_i seem to be a predictor. This is somewhat surprising since a highly activated immune system often is a bad sign. In all cases, the number of immune cells N_i is a better predictor than the number of tumor cells N_t. This confirms the hypothesis that the amount of immune cells are a better prognostic biomarker than the amount of tumor cells.

It is not clear if the spacial metrics $K(r)$, $N_i s(r)$, and $N_i t(r)$ carry predictive information or if they are just a proxy for the number of immune cells. The WSIs samples are only 1024 x 1024 pixels large. The special metrics will thus be distorted for all cells closer than 100

pixels to the edge. This could be addressed by increasing the patch size or reducing the radius parameter r.

The AUC for different number of WSI-patches is shown in Figure 16. The AUC-scores seem to increase somewhat with larger sample-sizes, but the returns diminish very fast, note that the x-scale is logarithmic. Some of the evaluations using only a few WSI-patches are as good as the ones using hundreds. This, and the extremely varied performance using few patches, suggest that different regions in a WSI carry much more predictive information than others. It would be interesting to investigate where important regions are located and in turn implement a smarter sampling method.

I think that the most important paths going forward would be:

- Investigating which regions of the WSIs that yield the best predictions.
- Adding interactions between well performing terms in the logistic regression model for outcome prediction.
- Classifying cells in segmentation training data in order to train a multi-class U-Net.

References

[1] Catherine Downs-Holmes and Paula Silverman. Breast cancer. *The Nurse Practitioner*, 36(12):20–26, 2011.

[2] Kartik Aysola Akshata Desai. Triple negative breast cancer – an overview. *Hereditary Genetics*, 2012.

[3] Lei He, L. Rodney Long, Sameer Antani, and George R. Thoma. Histology image analysis for carcinoma detection and grading. *Computer Methods and Programs in Biomedicine*, 107(3):538–556, 2012.

[4] R. Salgado, C. Denkert, S. Demaria, N. Sirtaine, F. Klauschen, G. Pruneri, S. Wienert, G. Van den Eynden, F. L. Baehner, F. Penault-Llorca, E. A. Perez, E. A. Thompson, W. F. Symmans, A. L. Richardson, J. Brock, C. Criscitiello, H. Bailey, M. Ignatiadis, G. Floris, J. Sparano, Z. Kos, T. Nielsen, D. L. Rimm, K. H. Allison, J. S. Reis-Filho, S. Loibl, C. Sotiriou, G. Viale, S. Badve, S. Adams, K. Willard-Gallo, and Sherene Loi. The evaluation of tumor-infiltrating lymphocytes (tils) in breast cancer: Recommendations by an international tils working group 2014. *Annals of Oncology*, 26:259–271, 2 2015.

[5] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 1 2015.

[6] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807–814, Madison, WI, USA, 2010. Omnipress.

[7] Robert Hecht-Nielsen. *Theory of the Backpropagation Neural Network**Based on "non-indent" by Robert Hecht-Nielsen, which appeared in Proceedings of the International Joint Conference on Neural Networks 1, 593–611, June 1989. © 1989 IEEE.* Elsevier, 1992.

[8] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[9] M. D. Zeiler, G. W. Taylor, and R. Fergus. Adaptive deconvolutional networks for mid and high level feature learning. In *2011 International Conference on Computer Vision*, pages 2018–2025, November 2011.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017.

[11] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[12] Le Hou, Rajarsi Gupta, John S Van Arnam, Yuwei Zhang, Kaustubh Sivalenka, Dimitris Samaras, M Kurc, and Joel H Saltz. Dataset of segmented nuclei in hematoxylin and eosin stained histopathology images of ten cancer types. *Nature*, 2020.

[13] N. Kumar, R. Verma, S. Sharma, S. Bhargava, A. Vahadane, and A. Sethi. A dataset and a technique for generalized nuclear segmentation for computational pathology. *IEEE Transactions on Medical Imaging*, 36(7):1550–1560, 2017.

[14] N. Kumar, R. Verma, D. Anand, Y. Zhou, O. F. Onder, E. Tsougenis, H. Chen, P. A. Heng, J. Li, Z. Hu, Y. Wang, N. A. Koohbanani, M. Jahanifar, N. Z. Tajeddin, A. Gooya, N. Rajpoot, X. Ren, S. Zhou, Q. Wang, D. Shen, C. K. Yang, C. H. Weng, W. H. Yu, C. Y. Yeh, S. Yang, S. Xu, P. H. Yeung, P. Sun, A. Mahbod, G. Schaefer, I. Ellinger, R. Ecker, O. Smedby, C. Wang, B. Chidester, T. V. Ton, M. T. Tran, J. Ma, M. N. Do, S. Graham, Q. D. Vu, J. T. Kwak, A. Gunda, R. Chunduri, C. Hu, X. Zhou, D. Lotfi, R. Safdari, A. Kascenas, A. O'Neil, D. Eschweiler, J. Stegmaier, Y. Cui, B. Yin, K. Chen, X. Tian, P. Gruening, E. Barth, E. Arbel, I. Remer, A. Ben-Dor, E. Sirazitdinova, M. Kohl, S. Braunewell, Y. Li, X. Xie, L. Shen, J. Ma, K. D. Baksi, M. A. Khan, J. Choo, A. Colomer, V. Naranjo, L. Pei, K. M. Iftekharuddin, K. Roy, D. Bhattacharjee, A. Pedraza, M. G. Bueno, S. Devanathan, S. Radhakrishnan, P. Koduganty, Z. Wu, G. Cai, X. Liu, Y. Wang, and A. Sethi. A multi-organ nucleus segmentation challenge. *IEEE Transactions on Medical Imaging*, 39(5):1380–1391, 2020.

[15] P. Naylor, M. Laé, F. Reyal, and T. Walter. Nuclei segmentation in histopathology images using deep neural networks. In *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)*, pages 933–936, 2017.

[16] Robert L. Camp, Veronique Neumeister, and David L. Rimm. A decade of tissue microarrays: Progress in the discovery and validation of cancer biomarkers. *Journal of Clinical Oncology*, 26:5630–5637, 12 2008.

[17] Yuxin Cui, Guiying Zhang, Zhonghao Liu, Zheng Xiong, and Jianjun Hu. A deep learning algorithm for one-step contour aware nuclei segmentation of histopathology images. *Medical and Biological Engineering and Computing*, 57:2027–2043, 9 2019.

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[19] B. D. Ripley. The second-order analysis of stationary point processes. *Journal of Applied Probability*, 13(2):255–266, 1976.

[20] Skipper Seabold and Josef Perktold. statsmodels: Econometric and statistical modeling with python. In *9th Python in Science Conference*, 2010.