UNIVERSITÀ DI PISA

DIPARTIMENTO DI INFORMATICA

Corso di Laurea Triennale in Informatica

TESI DI LAUREA

ANALYSIS OF WHOLE SLIDE IMAGES OF BREAST CANCER FOR PD-L1 DETECTION

Relatori:

Alina Sîrbu

Simone Bonechi

Candidato:

Giacomo Cignoni

ANNO ACCADEMICO 2020/2021

Abstract

This thesis aims to create and compare possible models to classify PD-L1 positivity of breast tumors based on high resolution whole slide image analysis. PD-L1 is a biomarker that appears in the form of recognizable brown stainings on the whole slide images of the tumour. Therefore the task consists in quantifying the stainings with respect to the entire tumour area, and then classifying the tumor as PD-L1 positive or negative.

A heuristic approach to classify pixels is pursued, based on thresholding the pixel color distance from a base color. This is used (1) to estimate the area of interest of the tumor and (2) to classify the PD-L1 positivity of the slide on the estimated area of interest of the tumor. Regarding only the estimation of PD-L1 positivity on the given area of interest (2), an alternative hybrid approach is proposed. This creates a histogram of color distances and applies machine learning models to it in order to classify the slide.

The models show promising results for tumor area estimation, PD-L1 classification using only color distance and thresholding and in most machine learning models for the alternative approach for PD-L1 classification.

Contents

Chapter 1

Introduction

PD-L1 is an specific biomarker that can be present in tumor tissue, important for indicating the tumor aggressiveness. High resolution digital scans of specially stained biopsies of tumors can highlight PD-L1 presence, recognizable by its characterizing brown colored staining. These scans, called Whole Slide Images (in short WSI), are the data on which a tumor can be diagnosed to be positve or negative to PD-L1 expression.

WSIs of tissue samples are extremely rich in information, mainly because of their high resolution, and diagnosis based on this information is an exclusive prerogative of trained pathologists or biotechnologists whose expertise is based on their own experience or training. Readouts are performed manually, and therefore they could easily be influenced by inherent bias, introduced by both visual limitations and cognitive traps. However, with the introduction of digital image analysis for WSIs, some of this information is amenable to more precise and, more importantly, reproducible extraction, which can reduce or potentially eliminate human bias[14].

In accordance with these principles, the main objective of this thesis is to create and compare models capable to classify WSIs relative to breast cancer based on their PD-L1 staining with a sufficient degree of reliability. These models does not aim to replace the experience of a pathologist, but to support their decisions providing reproducible and unbiased results.

Another important objective is to create a model pipeline open to future improvement with the availability of more data, with more precise targets and with the use of more advanced image analysis techniques.

The thesis is organised as follows:

- In chapter 2 we will explain in more detail the role of PD-L1 in tumors and the modus operandi of pathologist to detect it. Then some previous work on whole slide image analysis in breast cancer and PD-L1 scoring with automated models will be examined, in order to have a view of the state of the art in this field.

- In chapter 3, an overview and brief explanation of the machine learning models employed in the model will be presented.
- In chapter 4, a summary of the used environment and libraries for the thesis is shown.
- In chapter 5, a brief overview of the dataset and its subdivision is explained.
- In chapter 6, we describe in detail the models introduced for this problem and their subcomponents.
- In chapter 7, we perform comparisons and evaluation of the model performance.
- Chapter 8 concludes the thesis and presents future possible developments.

Chapter 2

Background on PD-L1 and literature review

2.1 PD-L1 role in breast cancer

PD-L1[24] (programmed death-ligand 1) is a transmembrane protein that has been hypothesized to play an important role in suppressing the adaptive harm of the immune system during particular events such as pregnancy, autoimmune diseases and other disease states. The binding of PD-L1 to the inhibitory checkpoint molecule PD-1, present on the membrane of many white blood cells such as lymphocytes, acts as a "brake" for the immune system.

In solid tumors, the PD-1/PD-L1 inhibitory reaction can be misused to silence the immune system by increasing the expression of PD-L1 on the tumor cell surface. This leads to tumors with large sizes, high proliferation and lower survival rate (in regard to breast tumors). For these reasons, an high intensity of PD-L1 makes the tumor potentially more dangerous and thus making PD-L1 an important biomarker for tumor detection and classification, also opening to the possibility to PD-L1-specific treatments.

2.2 PD-L1 staining and scoring methods

PD-L1 presence in a tumor is observed by analyzing a Whole Slide Image (WSI) of the tumor itself. A WSI is a scan of a conventional glass slide in order to produce a digital slide. The WSIs are generated by dedicated high resolution scanners and usually have a resolution in the range of hundreds or thousands of megapixels[23].

In the case of our study, in order to highlight the presence of PD-L1, the tissue samples of the WSIs have been stained with the VENTANA PD-L1 (SP142) Assay, an immunohistochemical assay used to recognize the PD-L1 protein, co-developed by Roche/Ventana Medical Systems, Inc. (Ventana) and Roche/Genentech. The resulting WSIs show evident brown colored stains in areas and cells which have a high PD-L1 presence as shown in Figure 2.2. These brown PD-L1 stains are the indicators used for our approach to identify PD-L1[13].

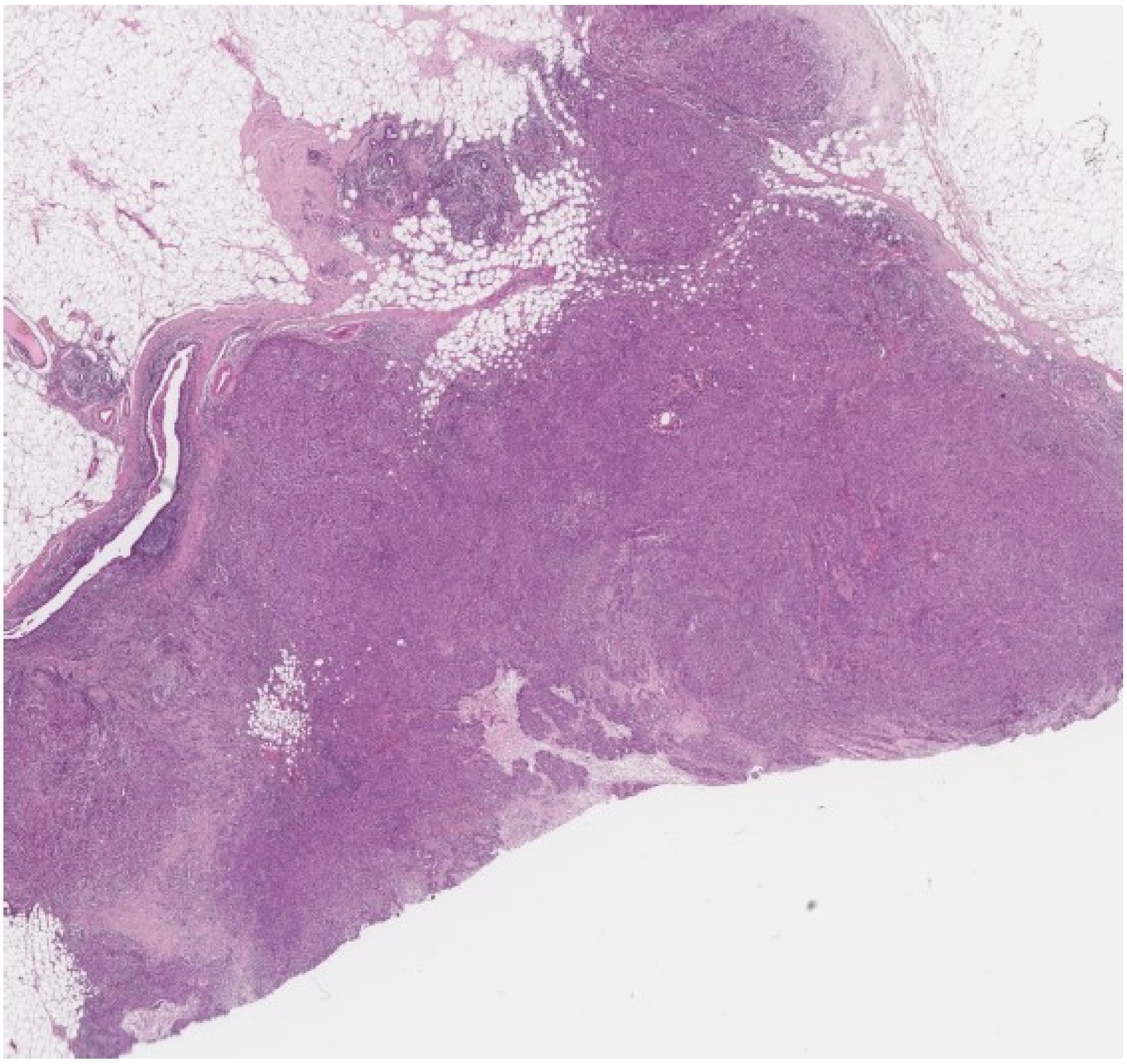


Figure 2.1: WSI of a breast tumor with standard hematoxylin and eosin staining

PD-L1 stains appear both in tumor-infiltrating immune cells (IC) and tumor cells (TC). The most common IC staining is the one to be considered for PD-L1 scoring; it usually presents in the form of aggregation of dark brown punctuate or linear stainings. The rarer TC stainings are not to be counted for PD-L1 scoring; it usually exhibits partial or complete circumferential membrane staining. It is more complex to differ-

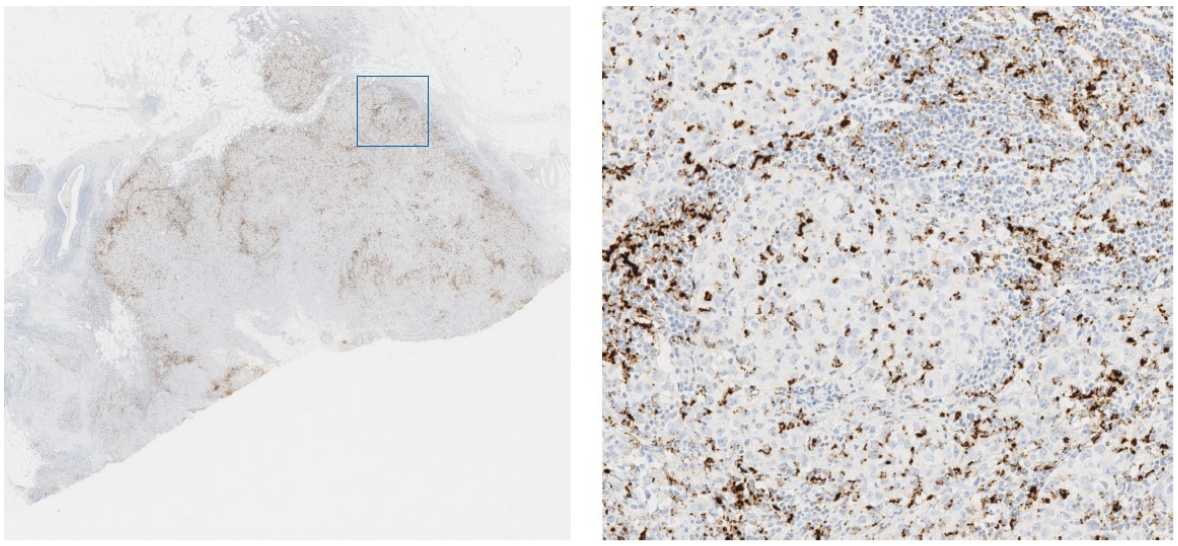


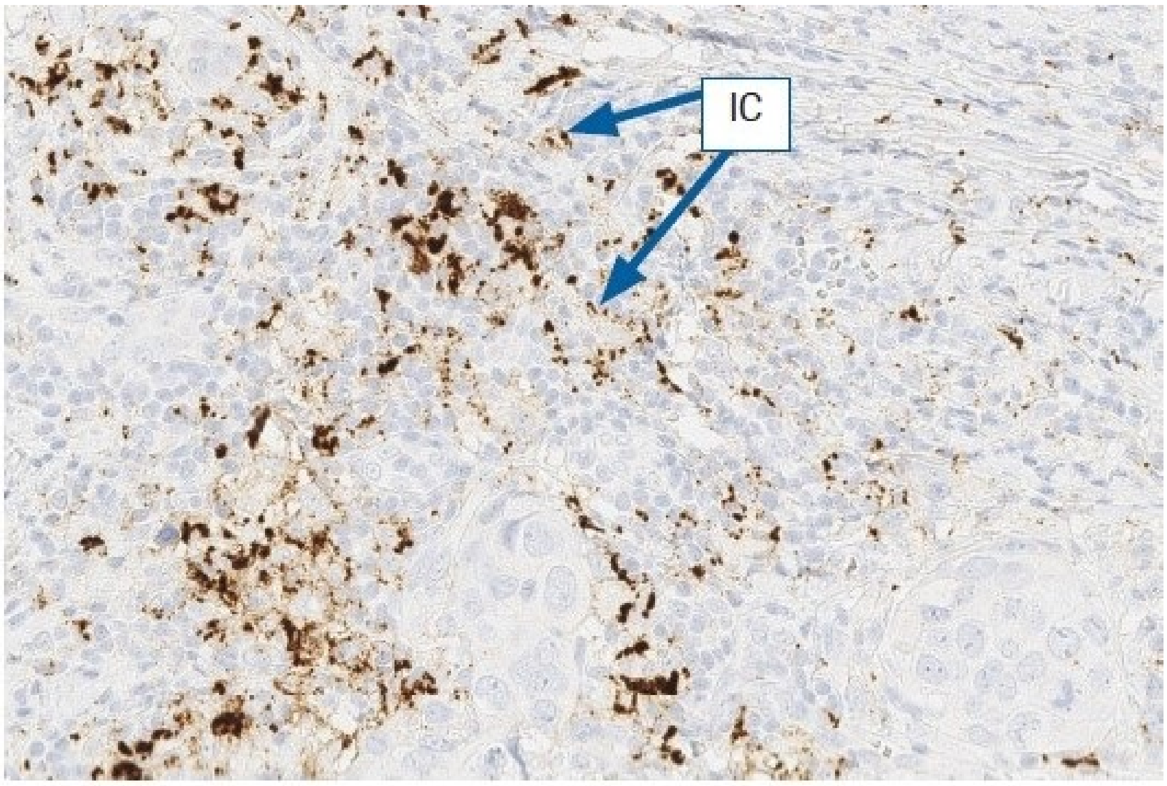
Figure 2.2: WSI of a breast tumor stained with the PD-L1 Assay to highlight PD-L1 presence. Next a zoomed in portion of the same WSI

entiate it from IC staining, often requiring to examine the standard hematoxylin and eosin (H&E) stained WSI for evaluation. The absence of cell-level annotations and of the H&E slides in our dataset, together with the relative less frequency of this kind of staining are the reason why TC stains is not used in the proposed PD-L1 scoring model.

PD-L1 Assay stained slides are classified by pathologists as having PD-L1 expression (in short, PD-L1 expression $\geq 1\%$ IC) if the specimen exhibits PD-L1 staining of any intensity in IC occupying $\geq 1\%$ of tumor area. If this condition is not satisfied for a WSI, it is classified as not having PD-L1 expression (in short, PD-L1 expression $< 1\%$ IC).

The individuation of the tumor area in a WSI, the ROI (abbreviated to ROI) over which to classify PD-L1 expression, is important for this classification and is a core problem for PD-L1 scoring.

Moreover, artifacts are present in whole slide images and can impact on a correct analysis of PD-L1 staining. They may be caused by errors in the staining process, such as blank spots (caused by static bubbles) or DAB spots, that are brown colored (formed by trapped DAB, a reagent used in the staining process). In this rare cases, to better asses the PD-L1 score, a new staining process is suggested to be taken to avoid the presence of artifacts during the analysis.



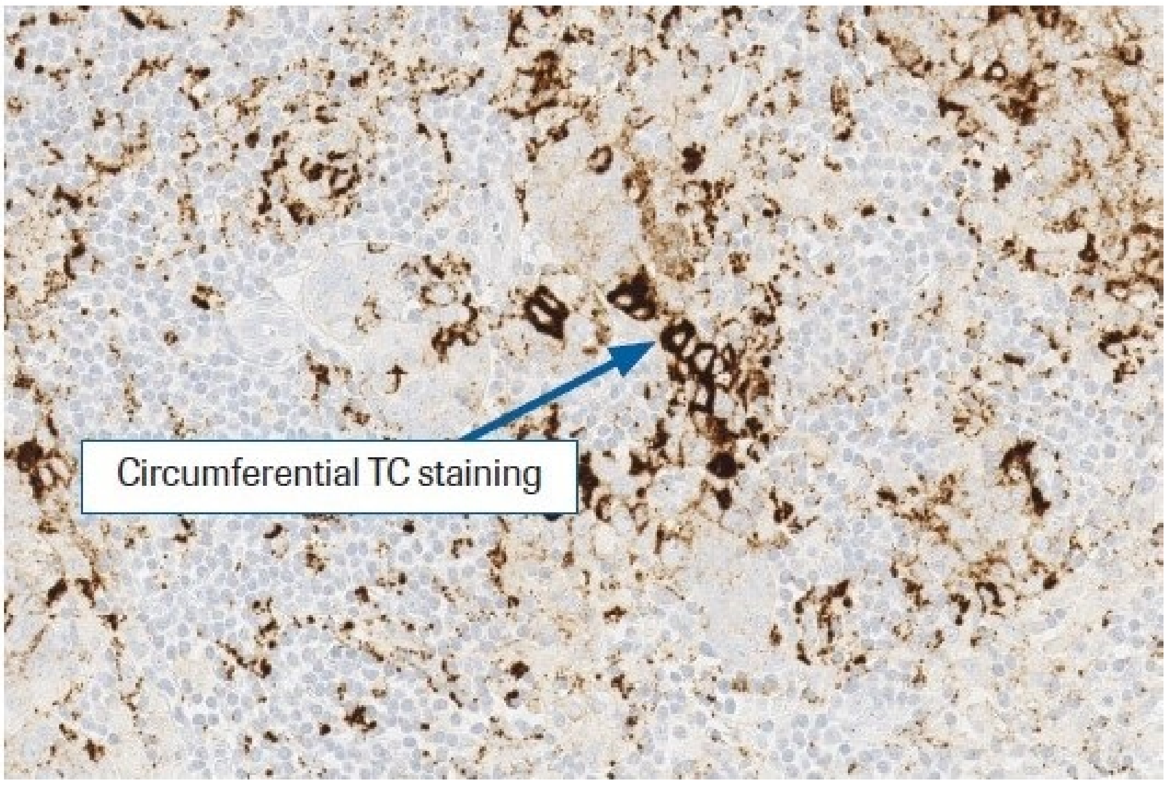
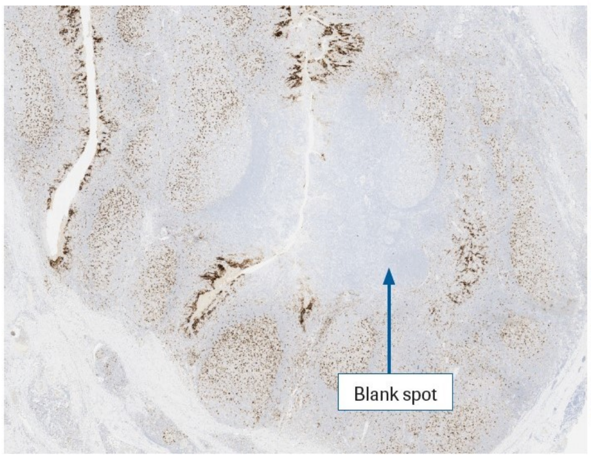
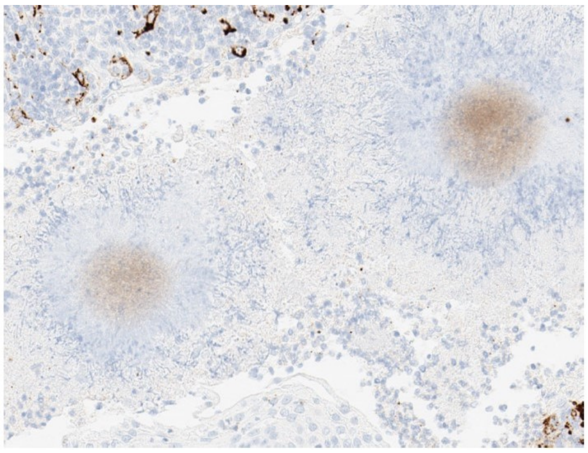


Figure 2.3: Portion of whole slide images showing the difference between IC staining (that should be considered for PD-L1 scoring) and TC staining (should not be related to PD-L1 scoring)



(a) Example of a blank spot staining artifact



(b) Example of 2 DAB spots, brown colored staining artifacts

Figure 2.4: Examples of staining artifacts that can impair the classification process

2.3 Literature review and state of the art for WSI analysis

Artificial Intelligence (AI) techniques are widely used in the medical image analysis field. Training an AI model as an inferential tool inherently lends itself toward assisting in diagnosis. Machine learning, one branch of AI, is growing in popularity in this field because of its capabilities of automated learning, i.e. models are exposed to a large amount of data with the expectation that they establish their own patterns to interpret and act on new data[14]. In particular, applications of Deep Learning techniques, most commonly Convolutional Neural Networks (CNNs) are having great success in tumor whole slide image analysis. [14].

2.3.1 CAMELYON17

An example of the adoption of AI and ML in medical image analysis is the CAMELYON17 challenge [2], organized by the Diagnostic Image Analysis Group and Department of Pathology of the Radboud University Medical Center. The purpose of this challenge was to come up with a fully automated method to identify breast cancer metastases in whole-slide images of lymph nodes and classify each lymph node into one of four stages (cancer negative, isolated tumor cells, micrometastases, or macrometastases) generating a patient-level prediction. The challenge is based on a large dataset of WSIs (1000 images) in common for all participants.

CAMELYON17 top performer

Given that the challenge is still open, we can analyse the current (November 2021) best performer. In their work[19], Sanghun Lee, Joonyoung Cho and Sun Woo Kim initially split the WSI in smaller patches and apply a deep learning model based on Deeplab v3+ under auto hard mining process (for each epoch, model studies whole patches of slides and chooses patches as train set whose intersections over union with annotated masks are less than 0.95).

Next, based on pixel-level results for whole slide images, slide-level heatmaps are generated and classified by rule-based criteria into the four possible stages. For this slide-level metastases classification, morphological features from heatmaps are extracted by using the DBSCAN algorithm. The model uses 3-fold cross validation and a separate test set and the mean slide-level accuracy on said test set is 0.93.

2.3.2 Deep learning approach for automated cancer detection and tumor proportion score estimation of PD-L1 expression in lung adenocarcinoma

This study[25] is of great relevance to our task, as it explores the use of AI techniques to estimate PD-L1 positivity (instead of classifying more generic lymph nodes) on a small dataset, similar in size to the one at our disposal, regarding lung instead of breast tumors. In particular, it makes use of deep learning to calculate the tumor proportion score (TPS) of PD-L1 for lung cancer patients.

The study used a relatively small but well annotated dataset, composed of 115 WSIs, divided in two similar sized sets, each stained with a different PD-L1 staining technique (22C3 and SP142). For each slide 3-5 cancer or stroma regions were selected to be annotated for building the model (171 regions of 22C3 and 178 regions of SP142). A Deep Convolutional Neural Network capable to estimate the TPS, was then trained on the data. Training was firstly carried out on the 22CR set (130 regions for training set and 41 for the test set). The model was then tested on both 22CR and SP142 sets. Then the same model was trained also on the SP142 set (103 regions for training) for fine tuning and tested again.

Changes in the loss function and sample weights allowed the authors to train two models, an high-specificity model and a more balanced one. Accuracy measures where obtained by comparing the regression scores from the models to pathologist slide classifications in a 14-point scale representing PD-L1 expression, ranging from ¡1% to 100%. The balanced one obtained a sensitivity of 90.09% and a specificity of 93.36% on the 22C3 set and a sensitivity of 90.63% and a specificity of 92.80% on the SP142 set after fine tuning on it.

2.3.3 Clinical-grade computational pathology using weakly supervised deep learning on whole slide images

The relevance of this article[16] consists in the use of weakly annotated WSIs, such as the ones at our disposal. Weakly annotated slides are much easier to obtain in large numbers, because each slide does not need to be annotated by a pathologist, which is time prohibitive at scale. Indeed, this study was able to retrieve a total of 44,732 weakly annotated slides of different tumor types, including breast cancer. The objective was to classify slides as positive ore negative, based on the presence or absence of tumor tissue.

To be more specific, the slide-level diagnosis casts a weak label on all tiles within a particular WSI. It is known that if the slide is negative, all of its tiles must also be negative and not contain tumor. In contrast, if the slide is positive, it must be true that at least one of all of the possible tiles contains tumor. This formalization of the WSI classification problem is an example of the general standard multiple instance assumption (MIL).

A two-step approach is used: first at tile level and subsequently at slide level. The MIL training procedure includes a full inference pass through the dataset, to rank the tiles according to their probability of being positive, and learning on the top-ranking tiles per slide via a CNN. Second, slide-level aggregation with a Recurrent Neural Network (RNN) occurs. The S most suspicious tiles in each slide are sequentially passed to the RNN to predict the final slide-level classification.

Accuracy over 95% was achieved for all tumor types. The study also emphasizes the importance of large datasets to obtain a good generalization capability in regard to tumor analysis.

Chapter 3

Overview of employed machine learning models

In this chapter an overview of 4 machine learning models is presented. We will use these models to classify WSIs as PD-L1 positive or negative in one of the explored pipelines.

3.1 Logistic Regression

Logistic regression[17, Chapter 4] is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of target in our case is dichotomous, as it is a binary classification problem. For the 2 target classes case (the dependent variable can assume the values (0, 1)), given k predictors $x_1, ..., x_k$, the general logistic function is the following composition of a sigmoid and a linear function:

$$p(x;\theta) = \frac{1}{1 + e^{-(b_0 + \sum_{i=1}^{k} b_i x_1)}}$$

where p is the probability of the dependent variable Y to be equal to 1 given x and $\theta = b_1, ..., b_k$ the parameter set on which the function is dependant on. This equation is equivalent to: $\log \frac{p}{1-p} = b_0 + \sum_{i=1}^{k} b_i x_1$, the logit function.

Logistic regression models are usually fit by maximum likelihood. Over the observation of N patterns, the log-likelihood has the form:

$$l(\theta) = \sum_{i=1}^{N} y_i \log p(x_i;\theta) + (1 - y_i) \log(1 - p(x_i;\theta))$$

The maximization can happen using various techniques, such as gradient descent, Newton method or limited memory BFGS. The latter was used in our case through the Scikit-learn library.

Various methods for regularization are possible and usually applied for logistic regression, in the Scikit-learn implementation[11] the inverse of regularization strength is determined by the C parameter.

3.2 SVM: Support Vector Machine

The support vector machine[18, Chapter 6] is a supervised learning classification algorithm. For a binary classification task, the concept on which the algorithm is based on is to construct a hyperplane as the decision surface on the input space, in a way that the separation margin between positive and negative examples is maximized. The optimal hyperplane is defined by $w_o^T x + b$, while the margin is defined by $\rho = \frac{2}{||w_o||}$. For N input training patterns x_i, with target class d_i, each one has to satisfy the zero classification errors constraint:

$$d_i(w^T x_i + b) \geq 1 \; \forall i = 1, ..., N$$

The support vectors (which give the name to the model too) are input pattern that lie exactly on the margin (the margin depends on those patterns), the previous constraint for those is exactly equal to 1.

Admission of errors (due to noise or other), i.e. patterns not correctly separated by the hyperplane with margin, is possible with the introduction of *slack variables*. The constraint is transformed into:

$$d_i(w^T x_i + b) \geq 1 - \xi_i, \; \xi_i \geq 0 \; \forall i = 1, ..., N$$

This formulation is valid for a linearly separable problem. To allow classification on non-linearly separable problems, we make use of kernels. Φ is a function that maps inputs x in an higher dimensional space in which pattern can be linearly separable $\mathbb{R}^{m_0} \rightarrow \mathbb{R}^{m_1}$, where m_0 is the initial feature space dimension and m_1 the higher dimensional feature space dimension. The hyperplane becomes: $w^T \Phi(x) + b = 0$

The kernel k consists in a function to calculate dot product in an higher-level feature space without needing to know Φ. $k : \mathbb{R}^{m_0} \times \mathbb{R}^{m_0} \rightarrow \mathbb{R}$ The kernel function is: $k(x_i, x) = \Phi^T(x_i)\Phi(x)$

The objective function in the primal form to minimize is:

$$\Psi(w, \xi) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i$$

with the new constraint $d_i(w^T\Phi(x_i)) \geq 1 - \xi_i,\ \xi_i \geq 0\ \forall i = 1, ..., N$ C is an hyperparameter inversely proportional to the tolerance of the SVM model to errors.

The objective function to maximize through the parameters α_i in the dual form is:

$$Q(\alpha) = \sum_{i=1}^{N}(\alpha_i - \frac{1}{2}\sum_{j=1}^{N}(\alpha_i\alpha_j d_i d_j k(x_i, x_j)))$$

satisfying the constraints:

$$\sum_{i=1}^{N}\alpha_i d_i = 0$$

$$0 \geq \alpha_i \geq C\forall i = 1, ..., N$$

The optimization problem is then solved through the dual form. Different kind of kernels can be used, such as RBF (Radial Basis Function) or polynomial of grade p.

A prediction of the model on a new x is calculated with $\sum_{i=1}^{N}\alpha_i d_i k(x, x_i)$

3.3 Decision Tree

Decision tree[22, Chapter 3] is a learning method for approximating discrete-valued target functions, in which the learned function is represented by a decision tree. Decision trees classify instances (patterns) by sorting them down the tree from the root to some leaf node, which provides the classification of the instance.

Each node in the tree specifies a check of some attribute, one of the feature, of the instance, and each branch descending from that node corresponds to one of the possible values for this attribute. An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute in the given example. This process is then repeated for the subtree rooted at the new node.

Each path from the tree root to a leaf corresponds to a conjunction of a sequence of attribute tests, while the whole decision tree represents a disjunction of these conjunctions.

Most algorithms for decision tree learning are based on the simpler ID3, an algorithm that employs a top-down, greedy search on the space of the possible decision trees.

A sketch of the algorithm is here briefly presented.

The best attribute is selected and used as the test at the root node of the tree. This choice by the attribute with the highest *information gain*, a statistical property of each attribute that measures of how well a given attribute separates the training examples according to their target classification.

A descendant of the root node is then created for each possible value of this attribute, and the training examples are sorted to the appropriate descendant node depending on the value assumed by the attribute. The entire process is then repeated using the training examples associated with each descendant node to select the best attribute to test at that point in the tree.

Derived algorithm include C4.5, C5.0 and CART. The Scikit-learn library[8] employed in this thesis uses an optimized version of the CART algorithm.

3.4 Random Forest

The random forest model[17, Chapter 15] extends the concept of bagging in regard to decision trees, building a large set of uncorrelated trees. The concept of bagging is to average many noisy but approximately unbiased models built upon subsets of the original training set, in order to reduce variance and help with overfitting. For a classification task, a majority vote is taken among all the models to select the output class.

In random forest, each decision tree T is built from a bootstrap sample of size N of the original data. Until n_{min} is reached, the minimum number of nodes of T, the following step is applied recursively on each terminal node: a random subset of features of dimension m are chosen along with the best variable among the m for maximizing the information gain, as a in standard decision tree; the node is then split in 2 daughter nodes.

In the Scikit-learn implementation of random forest[12], the number of features m is indicated by the parameter `max_features`.

Chapter 4

Tools and libraries

4.1 Python environment

Python 3.7 was the programming language of choice, due to its versatility and high number of libraries supported useful for our objective. To manage the Python environment in a simpler and more efficient way, I choose to use Anaconda[1], an open source package manager.

4.2 Jupyter notebooks and JupyterLab

The code for this thesis was written using JupyterLab[3], a web-based interactive development environment for Jupyter notebooks integrated inside Anaconda. A Jupyter notebook is an interactive computational environment, in which code execution, rich text, mathematics and plots can be combined to better expose and organize the problem and its solution.

4.2.1 Openslide

Openslide[7] is a C library that offers a simple interface to open whole-slide images saved in an ample set of formats. It allows reading a delimited portion of the WSI at the desired zoom level, avoiding to use excessive memory to open the entire gygabytes-sized WSI.

4.2.2 NumPy

NumPy[5] is a Python library that adds support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The core functionality of NumPy is its ”ndarray”, for n-dimensional array, data structure. Efficient operations on large ndarrays are possible with this library.

4.2.3 Matplotlib

Matplotlib[4] is a Python library that allows to draw graphs with a Matlab-like interface.

4.2.4 Pandas

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables. In our case its use was limited to interacting with the excel file of PD-L1 score results.

4.2.5 Scikit-learn

Scikit-learn[10] is a machine learning library for Python. It offers high-level functions to implement machine learning models and support function for validation and hyper-parameters selection.

4.2.6 Scikit-image

Scikit-image[9] (formerly scikits.image) is an open-source image processing library for Python. In this thesis scikit-image has been used to convert the image from RGB to CIELAB and to calculate the CIELAB color distance.

4.2.7 OpenCV

OpenCV[6] (Open Source Computer Vision Library) is a library of programming functions mainly aimed at real-time computer vision and image processing. In regard to this thesis it has been used to implement opening and closing morphological operations.

Chapter 5

Dataset

The available dataset consists of 40 WSIs available through the DROP project (Digital Research in Oncologic Pathology), funded by the SPARK programme at the University of Pisa. The images contain breast tumor regions with varying degrees of PD-L1 staining, various ROI sizes and different kinds of artifacts, and are labeled at the image level as PD-L1 positive or negative. An image in the dataset was found to be corrupted and for this reason, we decide to remove it from the dataset and use only 39 images for our experiments.

Each slide is saved in the `.tiff` format and has a full resolution in the range of approximately 50.000 to 150.000 pixels both in width and height. Each slide is accessible at 10 different level of resolution (0 to 9, 0 being the highest full resolution), each level having half the resolution of the previous one (a quarter of the pixels).

WSIs are identified by a strings of similar structure such as 20-COMP-075 or 19-COMP-004.

5.1 Dataset division

Before proceeding with the model, as part of data processing, the data set was randomly split in a training set (35 WSIs, 17 positive and 16 negative), used to tune the models, and a test set (6 WSIs, 3 positive and 3 negative), to evaluate the obtained models.

Chapter 6

Model

The objective of the base model is to create a pipeline capable of giving a good approximation of PD-L1 positivity for a given WSI. A full machine learning based approach could not be pursued as the provided data set was too small to properly train a model. For this reason we decided to use, as base model, a heuristic approach, and then enhanced it with the application of a ML model in the final step of the procedure. A full machine learning and especially deep learning model could probably lead to better performance but they require a significantly larger amount of training data. In the next phases of this project, if more samples or, maybe, a different kind of labels (e.g. pixel level annotation), become available we could consider to employ a model based on machine learning techniques.

The chosen heuristic approach is based on calculating for each pixel its color distance from a base color and, using a threshold on these distances, it is immediate to obtain the number of pixels below that threshold. This can be used to classify the entire WSI or single tiles (each subdivision of the partitioning of the WSI in equal disjoint square sub-images is called a tile). This approach was chosen because of its similarity to pathologists modus operandi in regard to PD-L1 estimations[13].

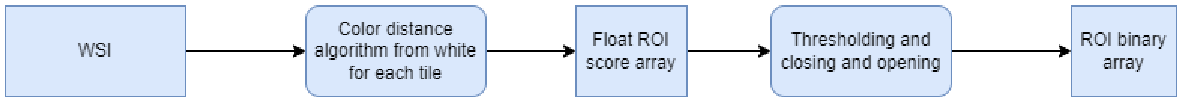
Unfortunately, given its simplicity, this approach is more prone to error in presence of artifacts compared to a full machine learning approach that could potentially learn from the data the presence of noisy elements inside the image.

As the PD-L1 percentage is calculated by pathologist as a percentage only on the area of the tumor (and not on the whole WSI), our model has 2 main objectives:

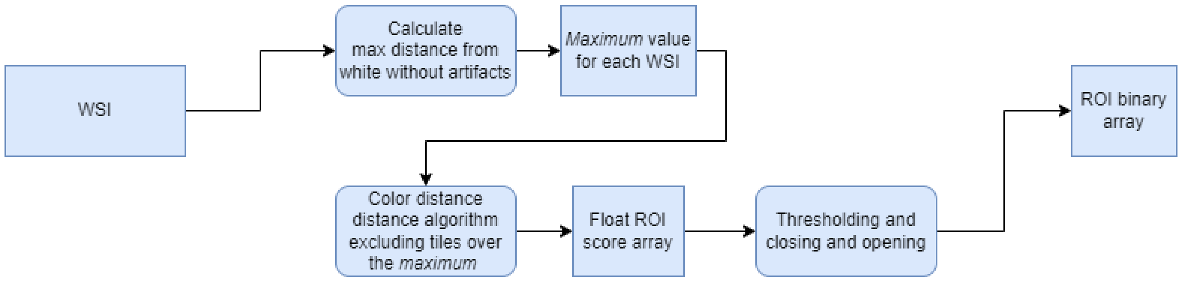
1. It needs to estimate and identify the area of the tumor, ROI, as accurately as possible

2. Then, the PD-L1 score needs to be computed only on the previously calculated ROI

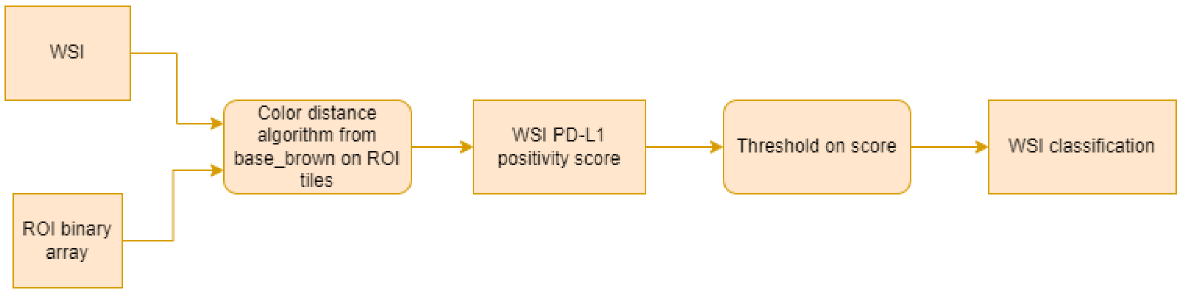
A heuristic approach based on color distance was used for both ROI identification and PD-L1 scoring classification; an hybrid pipeline comprehensive of machine learning models was also explored for PD-L1 scoring classification. Diagrams summarizing the explored model pipelines can be seen in figure 6.1.


WSI
Color distance algorithm from white for each tile
Float ROI score array
Thresholding and closing and opening
ROI binary array

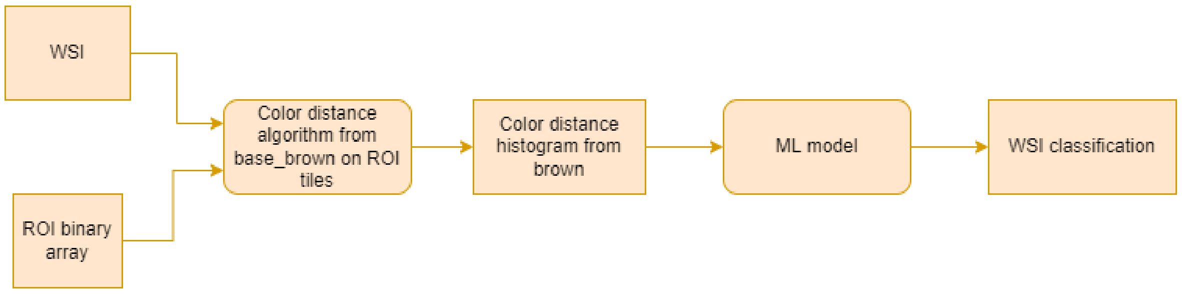
(a) Standard pipeline to calculate ROI


WSI
Calculate max distance from white without artifacts
Maximum value for each WSI
Color distance distance algorithm excluding tiles over the *maximum*
Float ROI score array
Thresholding and closing and opening
ROI binary array

(b) Pipeline to calculate normalized ROI with reduced artifacts


WSI
ROI binary array
Color distance algorithm from base_brown on ROI tiles
WSI PD-L1 positivity score
Threshold on score
WSI classification

(c) Pipeline of base model for PD-L1 scoring classification

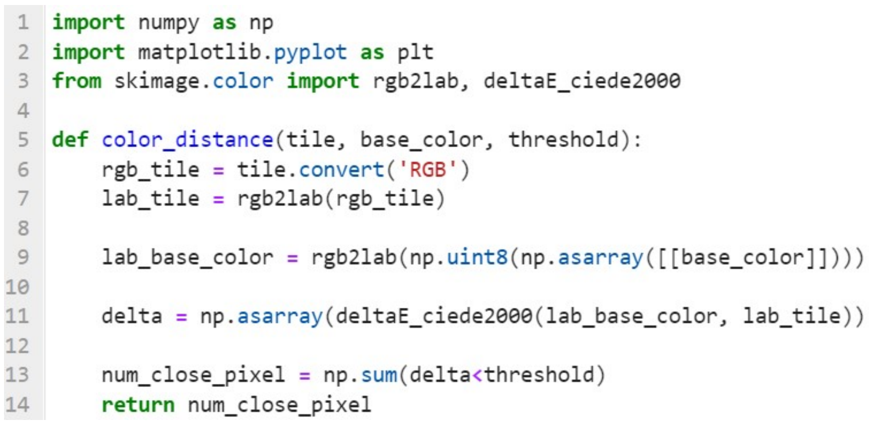

WSI
ROI binary array
Color distance algorithm from base_brown on ROI tiles
Color distance histogram from brown
ML model
WSI classification

(d) Pipeline of PD-L1 classification with ML models

Figure 6.1: Pipelines for ROI identification and PD-L1 scoring classification

6.1 Calculating color distance

The algorithm created for calculating the color distance from a certain color is defined by the function `color_distance`. The base version of this algorithm can be seen in figure 6.2. It takes as input a tile form the WSI, the base color expressed in RGB and an integer threshold, which represents the maximum distance below which a pixel is considered "close" to the base color. The function returns the number of "close" pixels.



```python
import numpy as np
import matplotlib.pyplot as plt
from skimage.color import rgb2lab, deltaE_ciede2000

def color_distance(tile, base_color, threshold):
    rgb_tile = tile.convert('RGB')
    lab_tile = rgb2lab(rgb_tile)

    lab_base_color = rgb2lab(np.uint8(np.asarray([[base_color]])))

    delta = np.asarray(deltaE_ciede2000(lab_base_color, lab_tile))

    num_close_pixel = np.sum(delta<threshold)
    return num_close_pixel
```

Figure 6.2: Portion of code representing the color_distance algorithm in its base form.

Both base color and the array representing the tile are converted to the CIELAB color space[15]. The CIELAB (or L*a*b) space is three-dimensional, and covers the entire range of human color perception. Because it is designed to approximate human vision, it is the best choice for our task, which consists in mimicking the naked eye analysis of a pathologist.

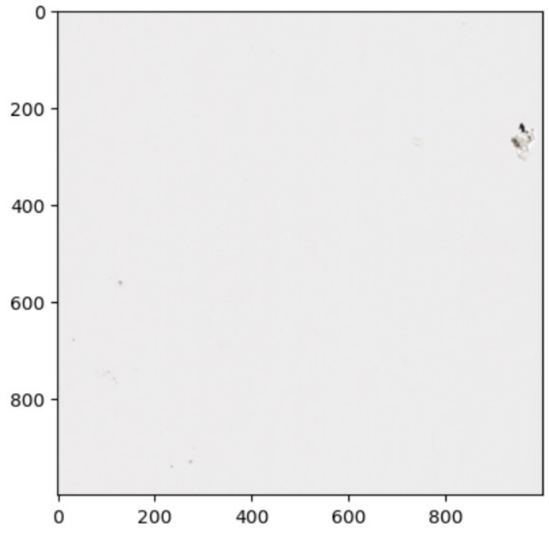
To calculate the distance in the CIELAB space, CIEDE2000 was used, the most recent and refined color difference formula defined by the CIE (International Commission on Illumination)[20]. It is implemented in Python using the library Scikit-image. Tile pixels are stored in Numpy array and manipulated with Numpy library methods for faster performance.

6.2 ROI identification

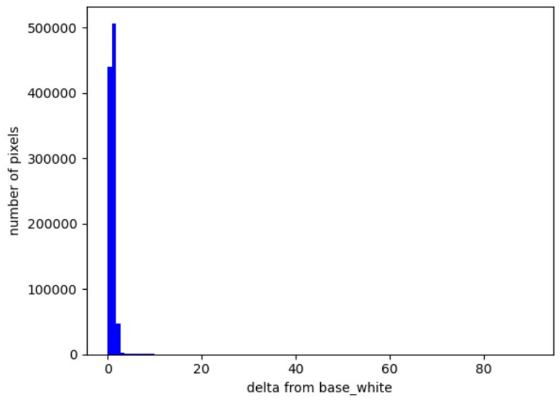
To identify the tumor ROI an approach based on color distance was used. We observed that the WSI areas outside the ROI predominantly have pixels of a particular shade of white (approximately RGB[238,238,238]). To test this hypothesis, the color distance algorithm was applied to a set of 5 tiles (1000x1000px, level 0) chosen clearly outside the ROI from different WSIs (an example is figure 6.3), using white (RGB[238,238,238]) as the base color. Applying a relatively low threshold of 5 on the color distance, we can easily observe from the first subtable in table 6.1 that over 99% of the pixels of each tile do not exceed the used threshold on the color distance. Repeating the procedure with tiles chosen inside the ROI (fig. 6.3), with same threshold and base color, we can observe that the percentage of pixels staying below the threshold is much lower. The same was repeated on the same tiles but with a lower resolution (level 2, 250x250px), with consistently faster computation time (only 1/4 of the pixels needs to be computed) but without significantly penalizing the results (as shown in the second subtable in table 6.1).

Pixels below white distance threshold in tiles *outside* ROI		
Tile (WSI and px coords)	Pixel percentage Level 0	Pixel percentage Level 2
19-COMP-011 (0,0)	99.99%	99.99%
19-COMP-012 (70000,32000)	100.00%	100.00%
19-COMP-017 (50000,0)	99.81%	99.85%
19-COMP-020 (80000,100000)	100.00%	100.00%
19-COMP-021 (35000,7000)	100.00%	100.00%
Pixels below white distance threshold in tiles *inside* ROI		
Tile (WSI and px coords)	Pixel percentage Level 0	Pixel percentage Level 2
19-COMP-004 (20000,50000)	57.82%	67.13%
19-COMP-011 (10000,73000)	76.22%	80.71%
19-COMP-014 (43000,40000)	48.03%	57.13%
19-COMP-020 (18000,85000)	83.77%	90.41%
19-COMP-021 (40000,70000)	78.51%	86.46%

Table 6.1: Results on color distance from white on tiles inside and outside the ROI

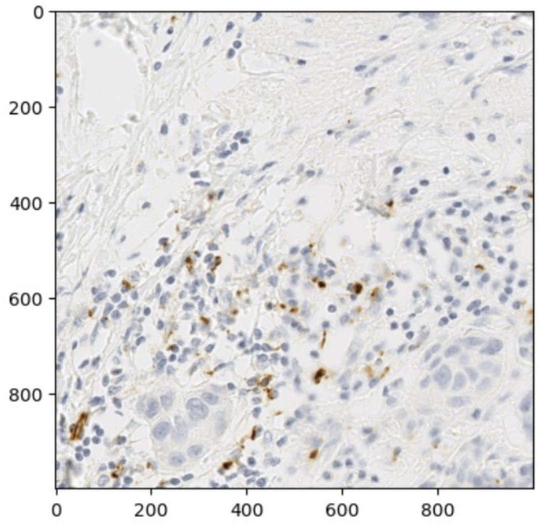


(a) Tile

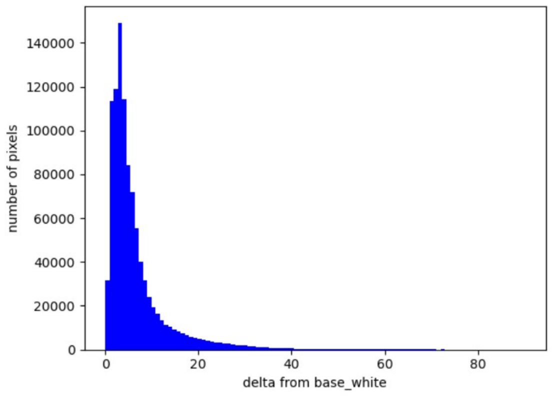


(b) Histogram of distance from white

Figure 6.3: Test of distance from white of tile *outside* ROI



(a) Tile



(b) Histogram of distance from white

Figure 6.4: test of white distance of tile *inside* ROI

Considering these results, an algorithm based on evaluating the color distance from white for each pixel of a tile to evaluate if the tile is part or not of the ROI seems promising. Of course, the algorithm would only provide an approximation of the ROI, especially on border tiles, in this case, choosing smaller tiles could improve the accuracy of the ROI identification.

6.2.1 ROI identification procedure for the entire WSI

The ROI-identification algorithm was implemented by using the `color_distance` algorithm described above.

Each WSI is divided in level 2 tiles (each level 2 pixel corresponds to 16 level 0 pixels), each of 128x128px size at level 2 (corresponding to 512x512px tiles at level 0). The inferior resolution allows the computation to be significantly faster, without much compromise on accuracy. Each tile is passed as a parameter to the `color_distance` algorithm, together with a color distance threshold of 5 and the base white color (RGB[238,238,238]). The output of the algorithm is the number num_w of pixels that are below the given distance threshold from the base white in a tile. Using this output and the total number of pixels in a tile we can compute the ratio of pixels "close to white" for each tile. The ratio for each tile is then stored in a bi-dimensional float array representing the tile positioning in the WSI. In more technical terms, for each tile of column x and row y (its position with respect to the WSI), the corresponding element of the array $R[y][x]$ contains a score s, $0 \leq s \leq 1$, with

$$s = \frac{num\_w}{tot\_tile\_pixels}$$

A higher s means the tile is more likely to be outside the ROI. This array is then saved in `.npy` format for later use. Figure 6.5 shows visually the results obtained for an example image.

6.3 Normalized ROI identification

In various slides of the training set there are present visual artifacts with darker, almost black, color that can impact on the ROI identification, as they are incorrectly classified as part of it. To prevent this, an enhanced technique to identify the ROI was created, taking into account outliers caused by artifacts. The rationale for the proposed procedure is based on the fact that the darker areas are homogeneous in color and have a large color distance from the base white.

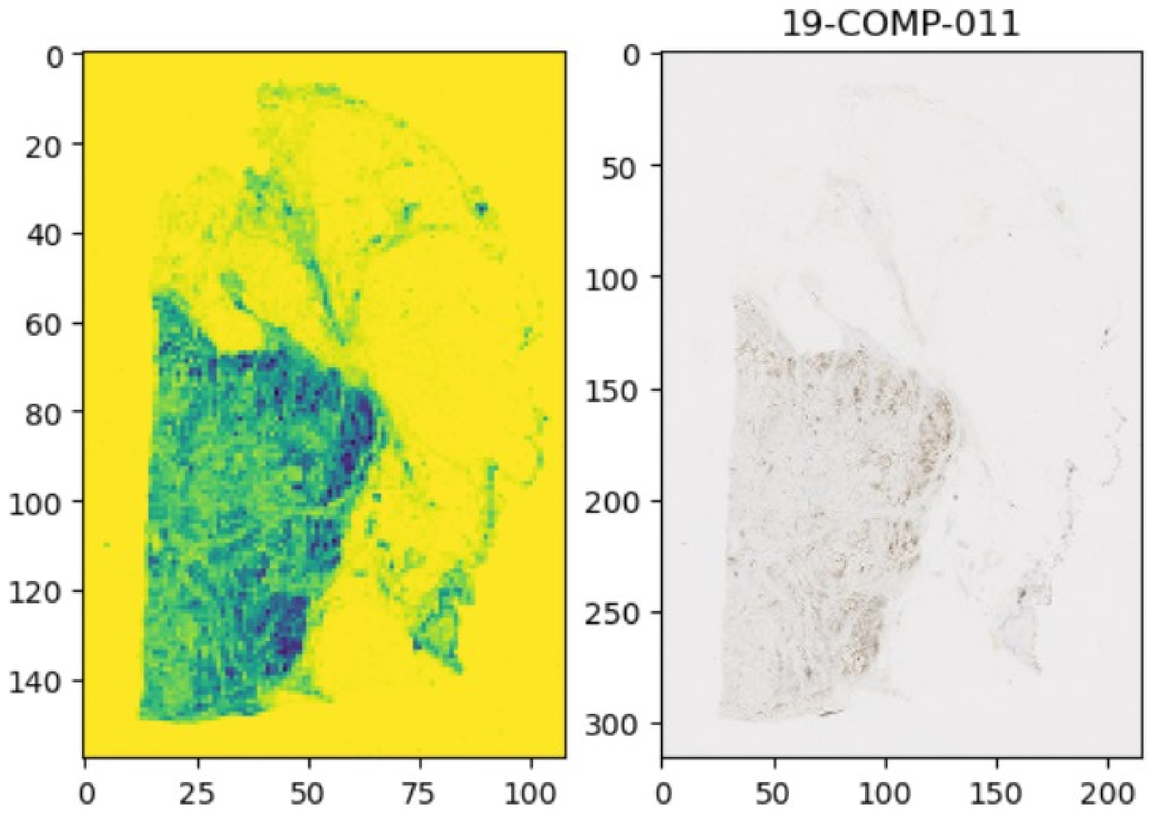


Figure 6.5: ROI identification in the form of float ROI score array (left) and the original WSI (right)

6.3.1 Calculating maximum distance without artifacts

Calculating the color distance histogram from base white for all WSIs is the preliminary step to find artifacts, which cause outliers in the histogram. To obtain the histogram, we use a new function that calculates the color distance of a tile, `color_distance_bins`. Instead of returning the number of pixels for which the distance from the base color is over a certain threshold, this algorithm returns the colour distance histogram for the pixels, i.e. the number of pixels in each bin of the histogram. The algorithm was applied to level 3 tiles, 512x512px, using a 100 bin histogram. Summing the results for each tile, bin per bin, we can obtain the histogram of color distance of the entire slide. Each histogram is normalized, i.e. it has a fixed range from 0 to 100 on the color distance.

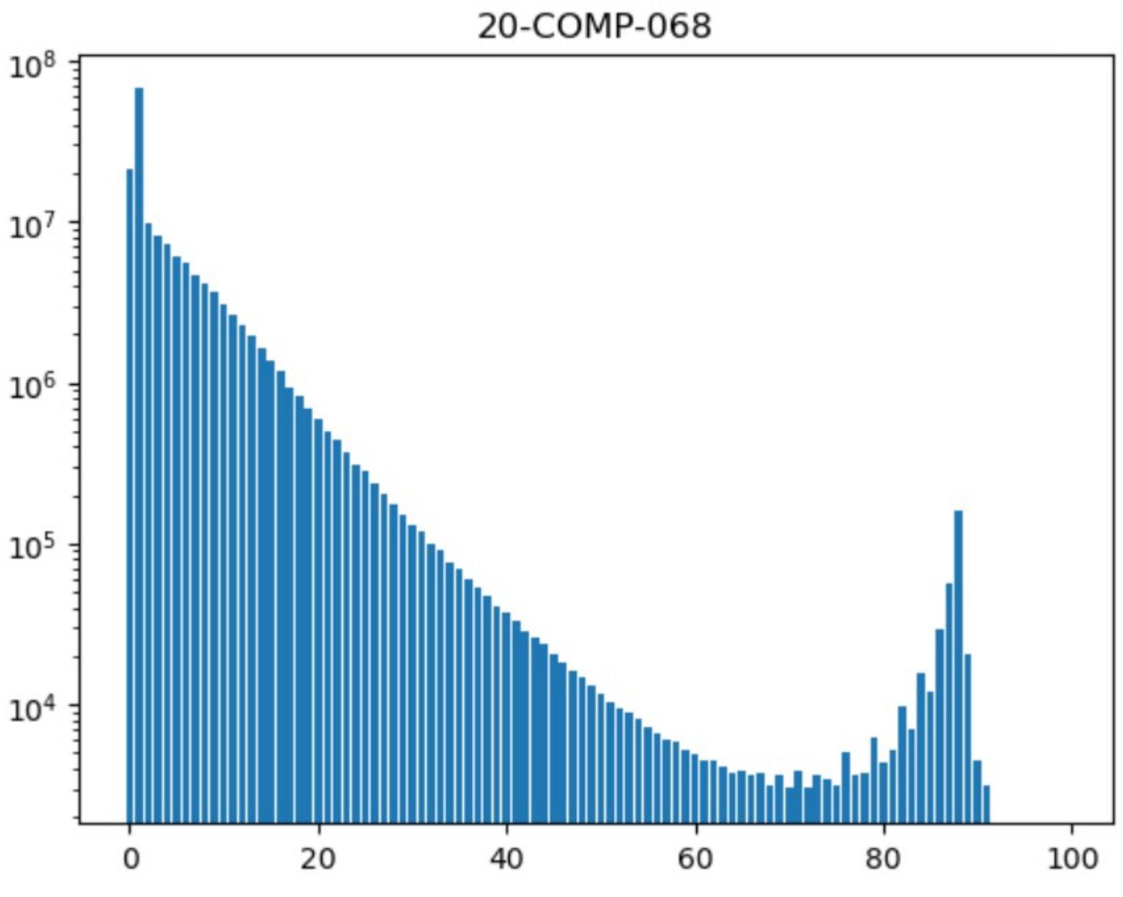


Figure 6.6: normalized histogram from slide 20-COMP-068 (shown in figure 7.4) with artifacts

Then, using the normalized histogram, it is possible to identify outliers in the bins. First the average and standard deviation over all WSIs of each bin of the histogram is calculated. Then, for each 50% upper portion of the slide histogram, values of bins which are considered outliers if $h[j][i] > 3 * std[i] + avg[i]$, where $h[j][i]$ indicates the $i - th$ bin of the histogram of $j - th$ WSI, while $avg[i]$ and $std[i]$ are calculated the average and standard deviation for bins in position i.

For each slide we select the outlier bin, if present, with lowest i index (closest to base white). We define as the *maximum* the color distance value of that bin, it is a cutoff value on color distance below which have not been identified any outliers (artifacts).

The *maximum* is calculated and stored for each WSI (WSIs with no detected outlier bin simply have 100 as the maximum, the superior delimiter of normalized color distance range).

6.3.2 Identifying ROI with reduced artifacts

After the previous computations, `color_distance_max`, a second variation of the `color_distance` algorithm, is applied to each tile (level 2, 256x256px tiles) of the WSI, to calculate color distance again from base white. As the original algorithm, it returns the number of pixels which distance from the base white is over the same threshold = 5, but if over 80% of the tile pixels have distance grater than the *maximum* of its tile, it marks the tile as an artifact, excluding it from the ROI (in practice, it returns the number of pixel of the tile, so that the ROI score s for the tile is 1).

The scores s and the resulting `.npy` array are calculated and stored same as in the non-normalized pipeline.

6.3.3 Float to binary ROI array

Before proceeding with the PD-L1 estimation, we have to get to a binary array which definitely identifies the ROI, obtained from the previously calculated float score array (both from the standard and normalized ROI identification). This includes a binary value for each tile, which flags whether the tile is part of the ROI or not. This is obtained by applying a threshold parameter, *ROI_threshold*: R being the float array and B the binary array,

$$B[y][x] = \begin{cases} 1 \; if \; R[y][x] < ROI\_threshold \\ 0 \; otherwise \end{cases}$$

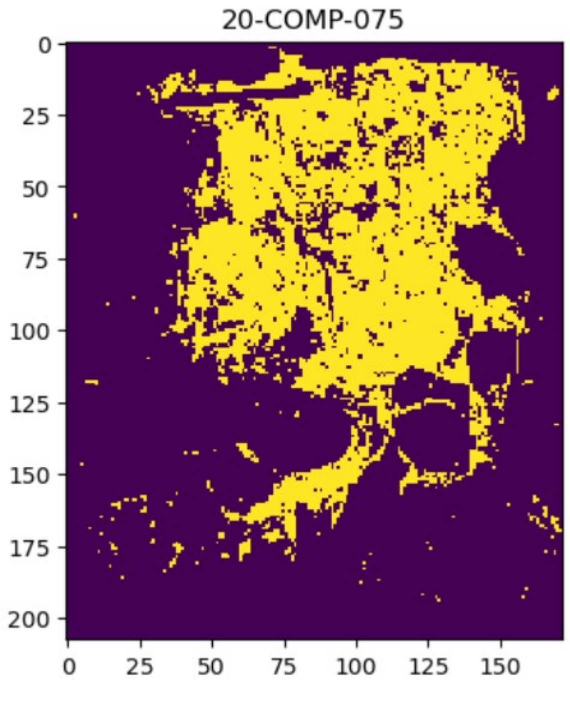
This process is applied both to the normalized and not normalized float ROI map, and the results saved in different arrays.

6.3.4 Closing and opening

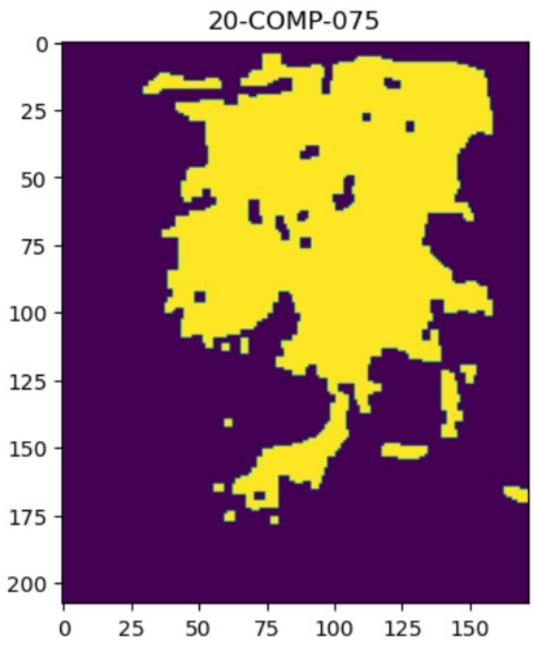
After obtaining the binary array, to obtain a smoother and more contiguous ROI, the morphological operations of closing and opening are applied to the binary array in this order.

The morphological operation of opening is defined as the dilation of the erosion of a set (binary image) A by a structuring element (in our case a square $3x3$ matrix of 1s) B. Similarly, the closing of a set A by a structuring element B is the erosion of the dilation of that set. Opening removes small objects from the foreground of an image, in our case it reduces or removes small "islands" of the ROI. On the other hand, closing removes small objects from the background of an image, in our case it reduces or removes "holes" in the ROI[21].

The concatenation of functions `dilate` and `erode` from the library OpenCV were used to apply the opening and closing morphological transformation. The resulting bidimensional binary array is again saved in a `.npy` array and used in later steps for PD-L1 detection. Figure 6.7 shows visually the results of this process.



(a) Before closing and opening



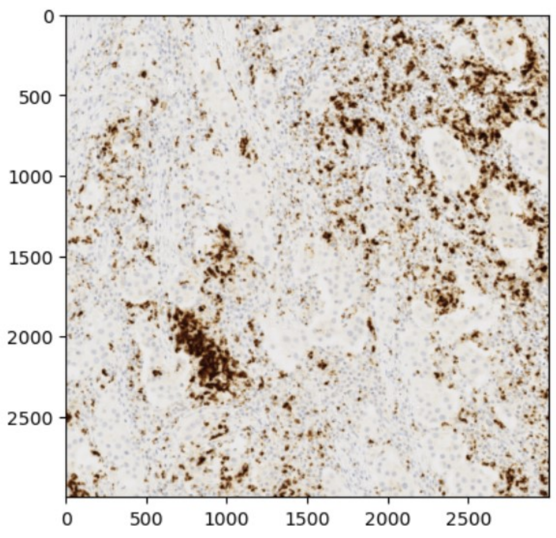
(b) After closing and opening

Figure 6.7: Effects of closing and opening on the binary ROI identification array

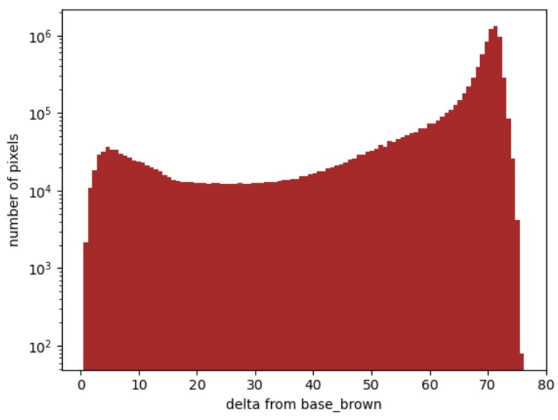
6.4 PD-L1 scoring

The regions of the WSI with high concentration of PD-L1 staining can be algorithmically identified using their brown colouring. To confirm this hypothesis, the `color_distance` algorithm was applied to a PD-L1 stained region and a not-stained region (both 3000x3000px, level 0) inside the ROI. We analyzed the color distance distribution of the pixels from the base color (a shade of brown RGB[106,60,33]) (figures 6.8 and 6.9).

It is easy to note that in the stained tile there is a significant amount of pixels ”close” to the base color, visible as a mode in the histogram, and it seems to be separable from the rest of the pixels by a threshold on the color distance. This bimodal distribution is not present in the PD-L1 negative tile.



(a) Tile



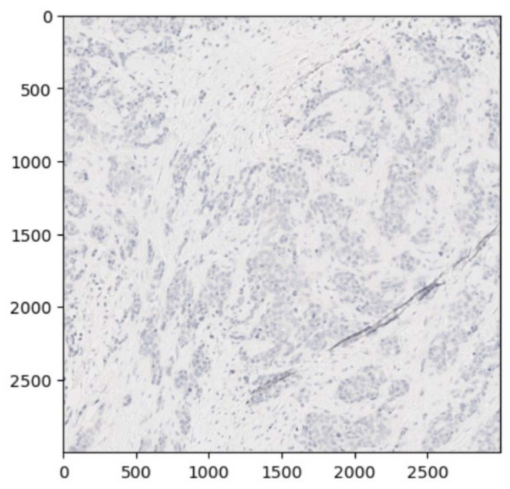
(b) Histogram in logarithmic scale

Figure 6.8: Color distance histogram from base_brown of a PD-L1 stained tile

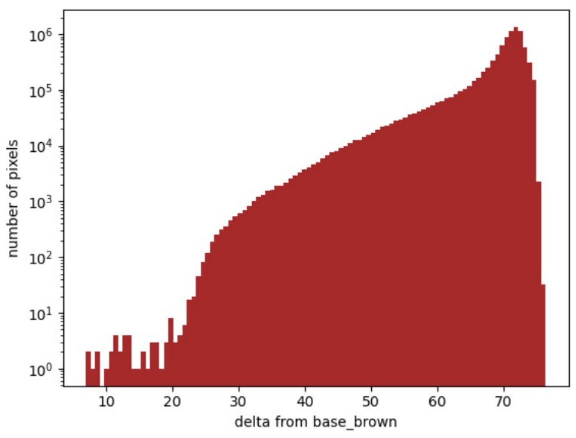
6.4.1 Heuristic algorithm to calculate PD-L1 score

The algorithm to calculate PD-L1 scoring for a whole-slide image is conceptually very similar to the one used to calculate the ROI. For each slide, we used the same tile subdivision as the one used for the ROI identification, as the PD-L1 percentage has to be calculated only on tiles belonging to it. The belonging of a tile to the ROI is determined by the binary `.npy` array previously saved.

The same `color_distance` algorithm is used only on the tiles classified as belonging to the ROI. The pixels belonging to a PD-L1 stained area have a relatively high color



(a) Tile



(b) Histogram in logarithmic scale

Figure 6.9: color distance histogram from base_brown of a *not* PD-L1 stained tile

variance, with multiple shades of brown. To estimate the shade of brown used as base color for the `color_distance` algorithm, a set of 16 pixels from different PD-L1 stains and different WSIs has been collected. The colors are converted to the CIELAB color space and then averaged to obtain the base color. The average color obtained, after converting it back to RGB, is [117.3, 88.9, 67.3] and it is referred as `base_brown`.

Then, each tile in the ROI is passed as an argument to the `color_distance` algorithm, together with `base_brown` and `brown_threshold`, the threshold on distance from `base_brown`. In the next chapter we will discuss results with different values for the `brown_threshold`. It returns n_t, the number of pixels whose distance from `base_brown` is below the threshold, the estimate of PD-L1 positive pixels for a tile t. To obtain a general positivity ratio, or score, for the entire WSI we sum the pixel estimate as positive for each tile and divide by the area of the whole ROI. In practice, if S is the score for the WSI and A the area of each tile,

$$S = \sum_{t \in ROI} \frac{n_t}{A}$$

The scores on all WSIs can then be visualized and a threshold on the score, `score_threshold`, is automatically set to classify correctly the most WSIs in the training set, based on the fact that a positive slide should be above `score_threshold` and the positive one below. A visual example of this threshold is shown in figure 7.8

A grid search over the training set was executed over the 2 possible parameters of the model: `ROI_threshold`, the threshold on the ROI float array (with possible values of 0.85, 0.90 and 0.95) and `brown_threshold`, mentioned in this section (with possible values of (9, 13, 18, 25). The grid search was repeated for the ROI calculated both with the standard and the normalized method.

6.5 Machine Learning models

Although a full machine learning model taking the entire slides as inputs was not employed due to the limited amount of data, we explored the use of ML models taking as inputs the histogram of the color distances from `base_brown` as an alternative way to classify PD-L1 positivity.

In more detail, the tiles belonging to the ROI are again identified by the `.npy` binary array initially calculated, using the artifact removal process.

To each slide belonging to the ROI the `color_distance_bins` algorithm is applied, passing `base_brown` as the base color to generate a normalized histogram with k bins for each tile (level 2 tiles, same dimensions as the ROI array tiles). All the tile histograms are then summed bin per bin, as done in section 5.3.2. The resulting histogram is the WSI normalized k-bins histogram of color distance from `base_brown`. An example of the histogram is shown in figure 6.10.

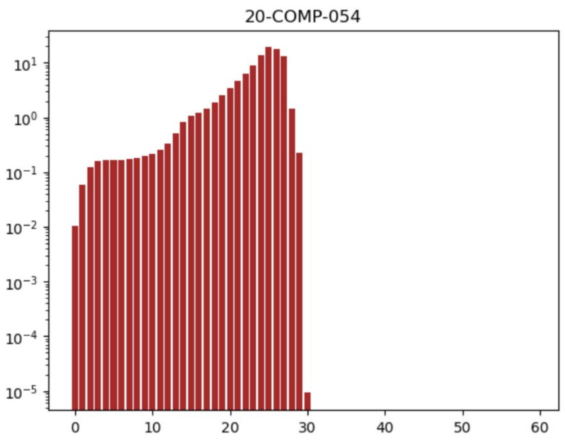
All of the WSIs histograms, i.e. the number of elements in each of the k bins, are saved in a bidimensional `.npy` array.

The saved histograms are then used as input for training the machine learning models, with the features for each image being the number of elements in each bin. Therefore, each model has k input features, one for each bin. The possible advantage of using this method instead of the heuristic above is that the ML model is able to learn the histogram shape that maximising accuracy, replacing the need for an empirically chosen and less flexible `brown_threshold`.

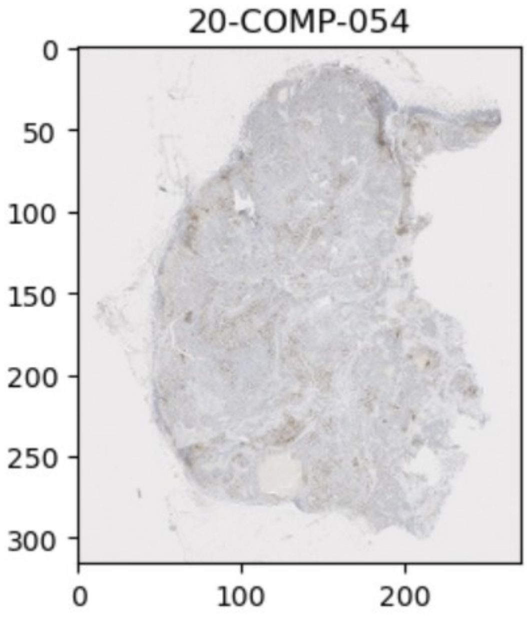
The chosen machine learning models are random forest, SVM, decision tree and logistic regression. Hyperparameter tuning on each model is done by an exhaustive grid search, using a 6-fold cross validation[17, Chapter 7] approach to better exploit the limited training set. After the best set of hyperparameters for each model has been found, that model is then retrained on the whole training set with those hyperparameters.

The number of bins k of the histogram and the `ROI_threshold` on which the histogram was calculated are hyperparameters common to all the 4 models. Each model then has its own model-specific hyperparameters. Table 6.2 shows the list of values of the hyperparameters.

The models, the grid search and the cross validation are all implemented using the



(a) Histogram in logarithmic scale



(b) Original slide

Figure 6.10: Normalized histogram of color distance from base_brown of slide 20-COMP-054 with 60 bins

Scikit-learn library.

Note: we did not explore any hyperparameters for our decision tree implementation.

Common ML hyperparameters	
Threshold on ROI	0.85, 0.90, 0.95
Number of histogram bins	40, 60, 100
SVM-specific hyperparameters	
kernel	rbf, linear, sigmoid, polinomial (degree 2,3,4)
gamma	scale, auto
C	0.1, 1, 10, 100, 300, 1000
Random forest-specific hyperparameters	
bootstrap	True, False
max_depth	100, None
max_features	square root, log2
min_samples_leaf	1, 2
n_estimators	100, 500, 1000
Logistic regression-specific hyperparameters	
max_iter	500, 1000, 2000
C	0.1, 0.2, 0.5, 1

Table 6.2: Hyperparameters on which the ML models where trained

Chapter 7

Results

7.1 ROI identification

The standard ROI identification process returns a quite good approximation of the tumor ROI for most WSIs. In the following some obtained results are reported. An example is given in figure 7.1 using a slide and its ROI array maps generated applying the 3 tested ROI thresholds (0.85, 0.90, 0.95).

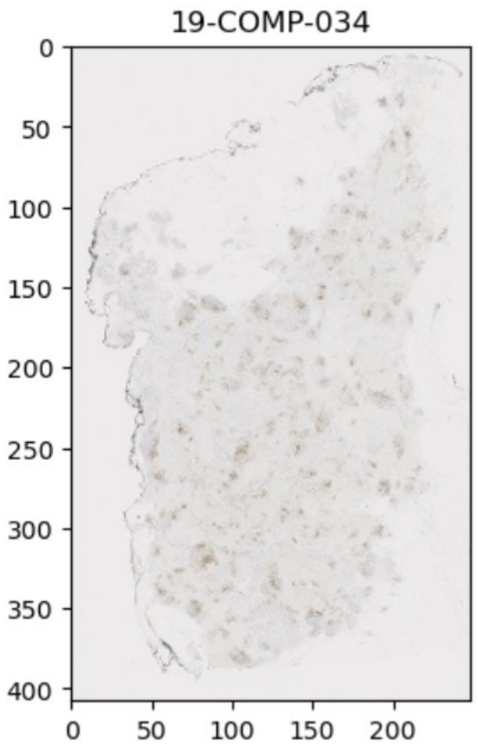
In some slides with a light tumor area, relatively close to white, the algorithm is not capable of identifying clearly a ROI. The slide in which this phenomenon is most present is slide 19-COMP-020 (figures 7.2 and 7.3), where changes in the threshold greatly vary the extension of the identified ROI.

Another kind of problematic ROI identification is caused by dark, relatively large artifacts, which, for their greater distance from white, are not excluded from the ROI. Slide 20-COMP-78 contains two extensive artifacts in the bottom corners which are incorrectly classified as being part of the ROI as shown in figure 7.4.

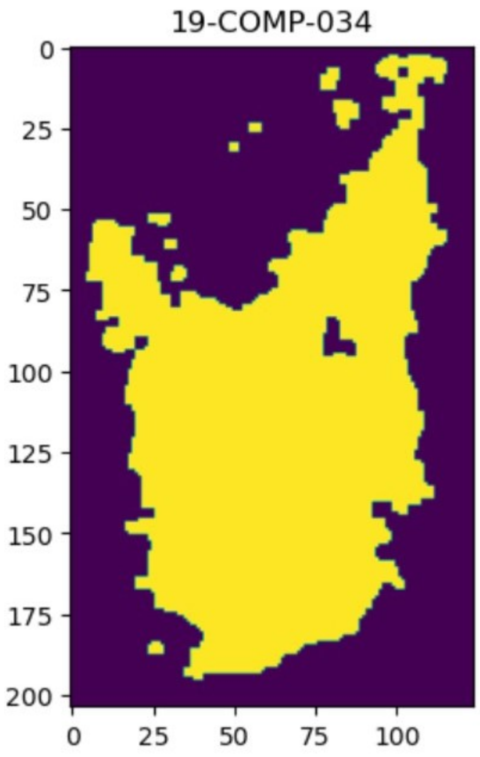
7.2 Normalised ROI identification

The improvements made in the ROI identification process to to decrease the impact on artifacts do not alter the capability of the model to correctly classify areas clearly part of the ROI.

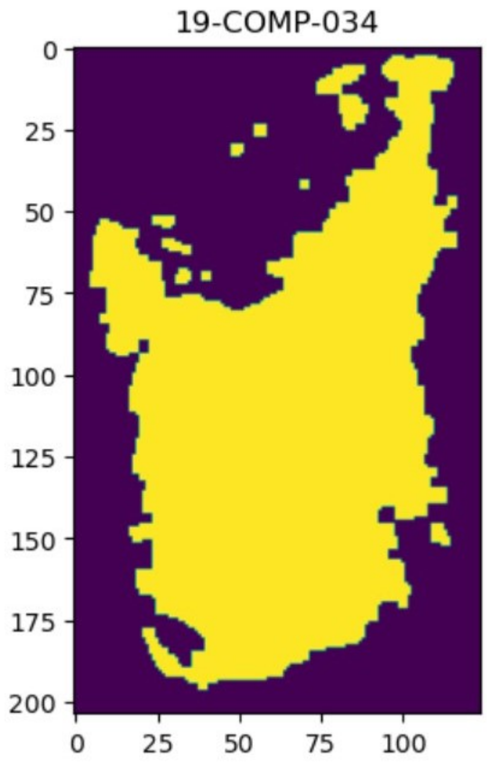
The exclusion of artifacts is effective mostly for the darker and most uniform ones. On the other hand, effectiveness of this method is limited for lighter coloured and faded artifacts, as their tiles rarely have most of the pixel color distance from white above the found *maximum* for the slide.



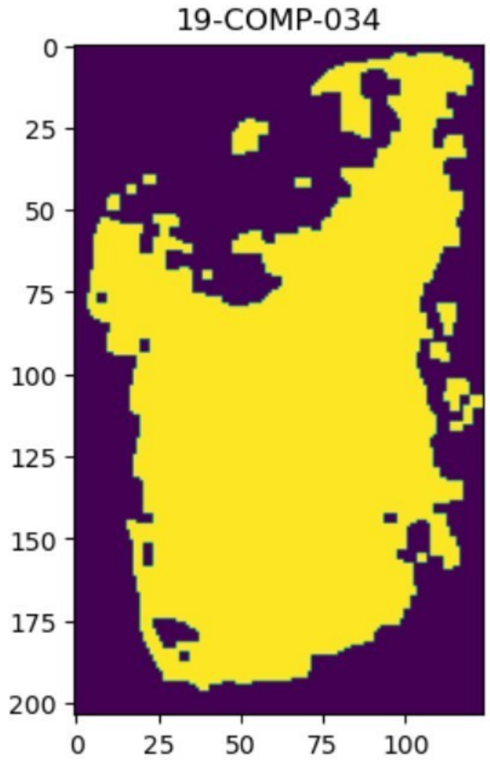
Original WSI



ROI map with threshold 0.85



ROI map with threshold 0.90



ROI map with threshold 0.95

Figure 7.1: Confrontation of original WSI and its ROI array maps after closing and opening and different thresholds

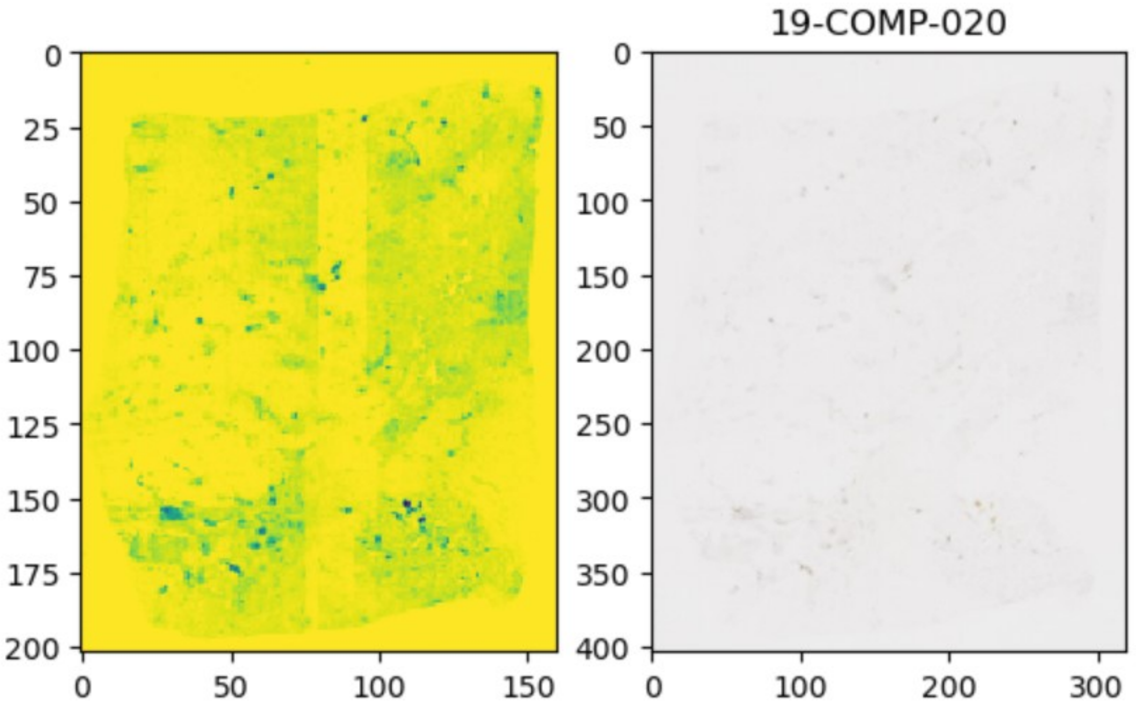
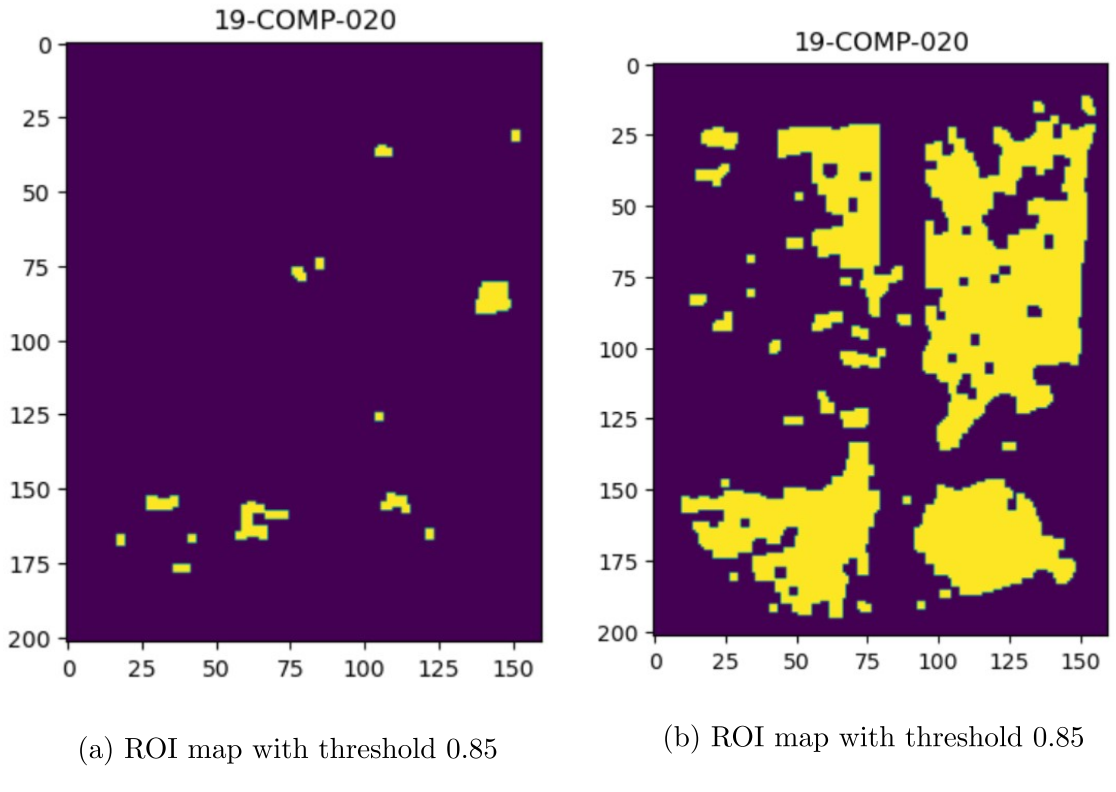


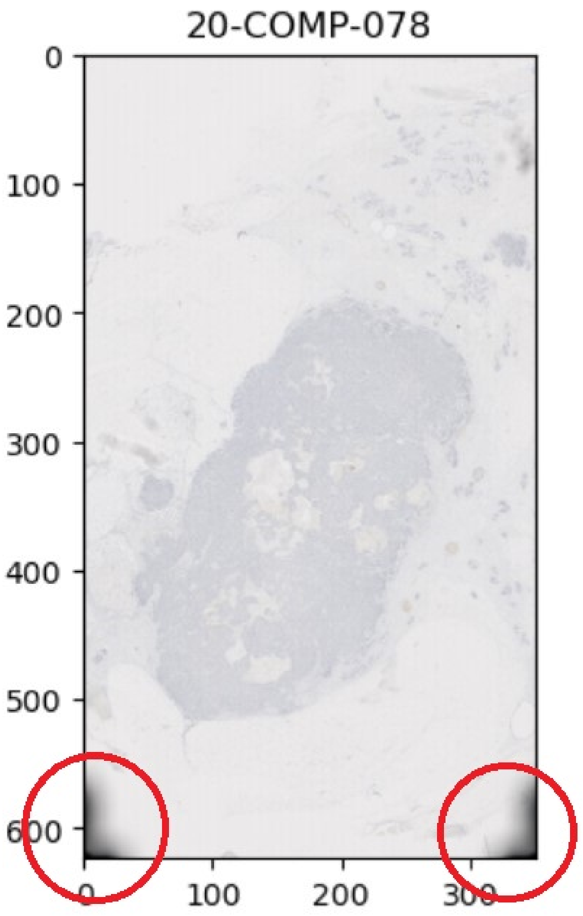
Figure 7.2: Very light WSI (right) and its ROI float array(left)



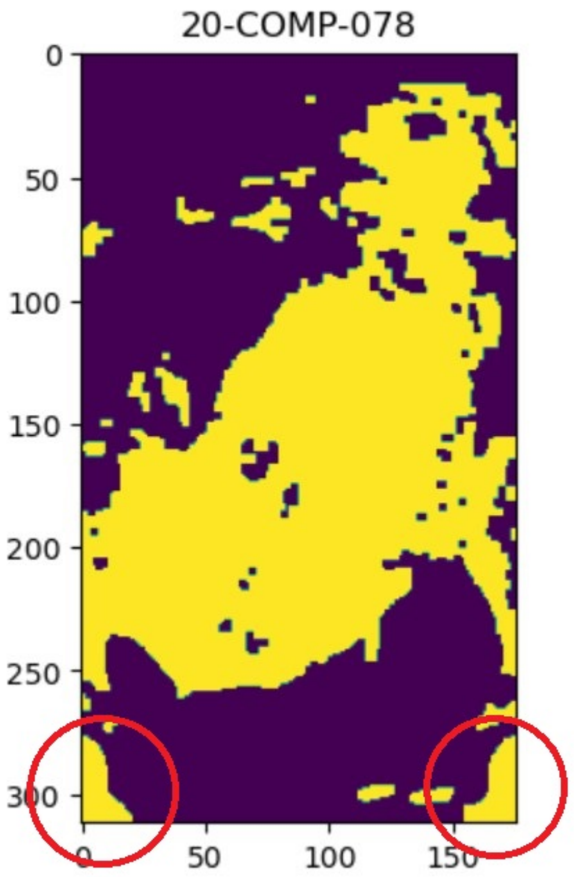
(a) ROI map with threshold 0.85

(b) ROI map with threshold 0.85

Figure 7.3: ROI map of a very lightly coloured WSI with two different thresholds applied



(a) Original WSI



(b) ROI map with threshold 0.85

Figure 7.4: WSI with highlighted corner artifacts and its ROI array map with artifact areas erroneously inserted inside the ROI

A successful exclusion of an artifact can be observed in slide 20-COMP-068, in which 2 artifacts are eliminated from the ROI map, as shown in figure 7.5.

7.3 Grid search for base model

The results of the Grid search executed on a standard and normalized ROI map can be seen in Figures 7.6 and 7.7. These show that all models showed a number of errors between 5 and 8, each corresponding to a misclassified slide.

On average, the results calculated using the non normalized slide have an higher number of errors. Also a lower threshold on the distance from `base_brown` (`brown_threshold`) leads to fewer misclassifications. Instead, changing the `ROI_threshold` seems to have only a slight impact on the number of errors, with marginally lower number of errors for a lower `ROI_threshold`.

Multiple combinations of parameters allow to obtain a minimum number or errors of 5 on the training set. The model based on the normalized ROI was preferred due to its consistency in having overall lower error and its apparent artifact reduction capabilities. Also a threshold on the ROI map of 0.85 was chosen for its relatively lower score in regard to the use with the normalized ROI.

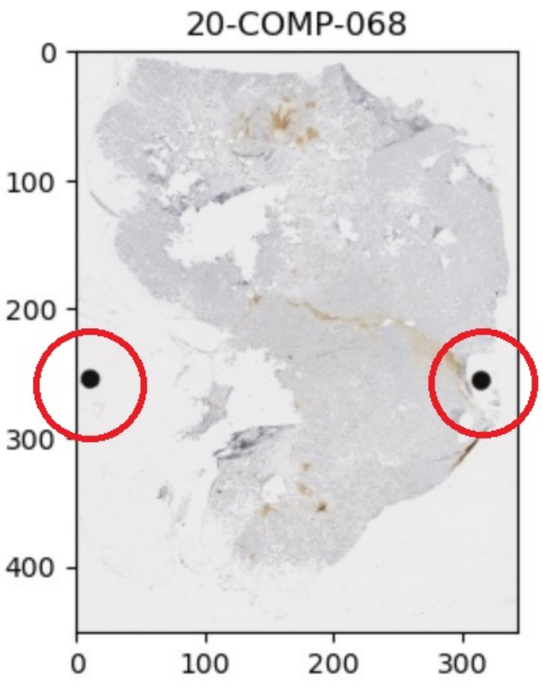
For these chosen parameters, both for a `brown_threshold` of 9 and 13 only 5 misclassifications are present. The `brown_threshold` of 9 was chosen because the 5 misclassified slides were more equally distributed between positive and negative ones (2 false negatives and 3 false positives), while `brown_threshold` of 13 showed a bias towards positive classification, with all 5 errors being false positives.

All in all, the selected base model for PD-L1 detection has the following parameters:

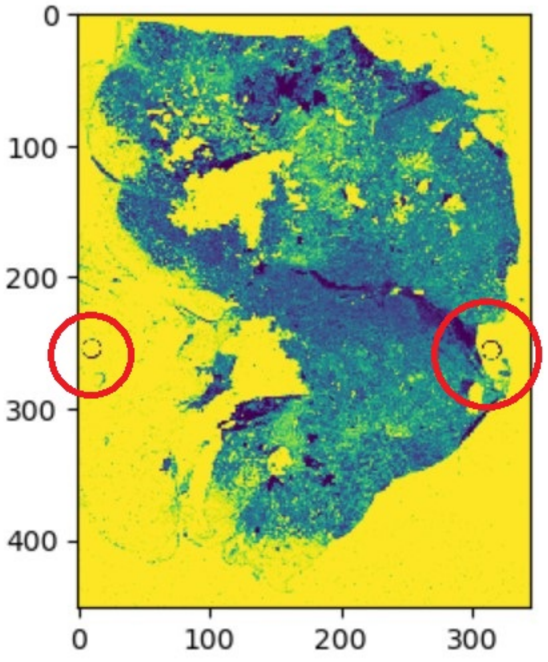
1. normalized ROI map
2. `ROI_threshold` on the float ROI map of 0.85
3. `brown_threshold` on color distance from `base_brown` of 9
4. `score_threshold` of 0.045%

Having 5 errors over a training set of 33 slides, it has an accuracy of 84.85%, and an F-score (F_1 score) of 85.71%.

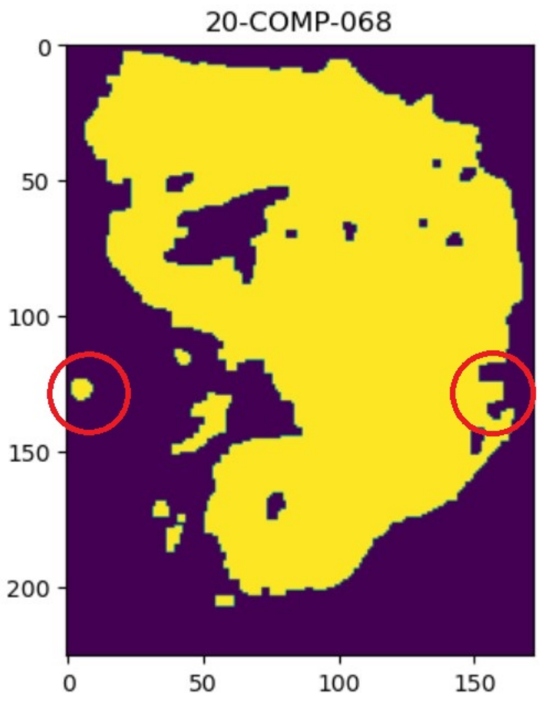
The classification results for the selected base model are shown in figure 7.8. The 2 false negatives in the training set of the selected base model show low levels of PD-L1



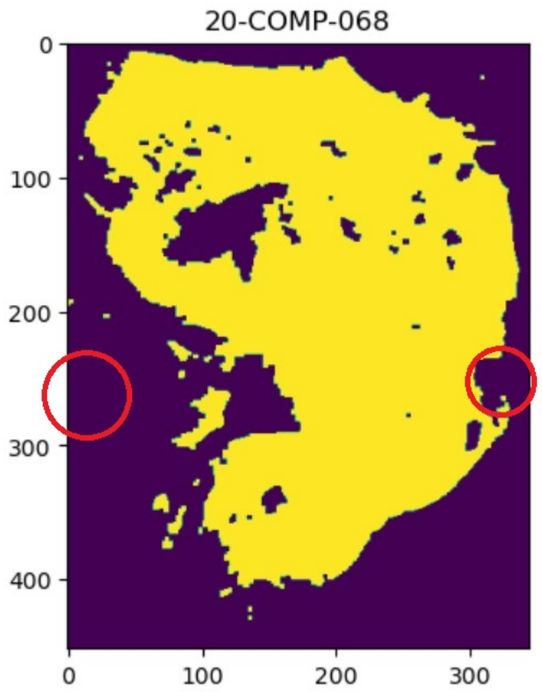
(a) Original WSI



(b) ROI float array after normalization has been applied



ROI map with threshold 0.85 from ROI identification *without* normalization.



ROI map with threshold 0.85 from *normalized* ROI identification.

Figure 7.5: WSI with 2 dark artifacts and confrontation with resulting ROI from standard and normalized ROI identification processes

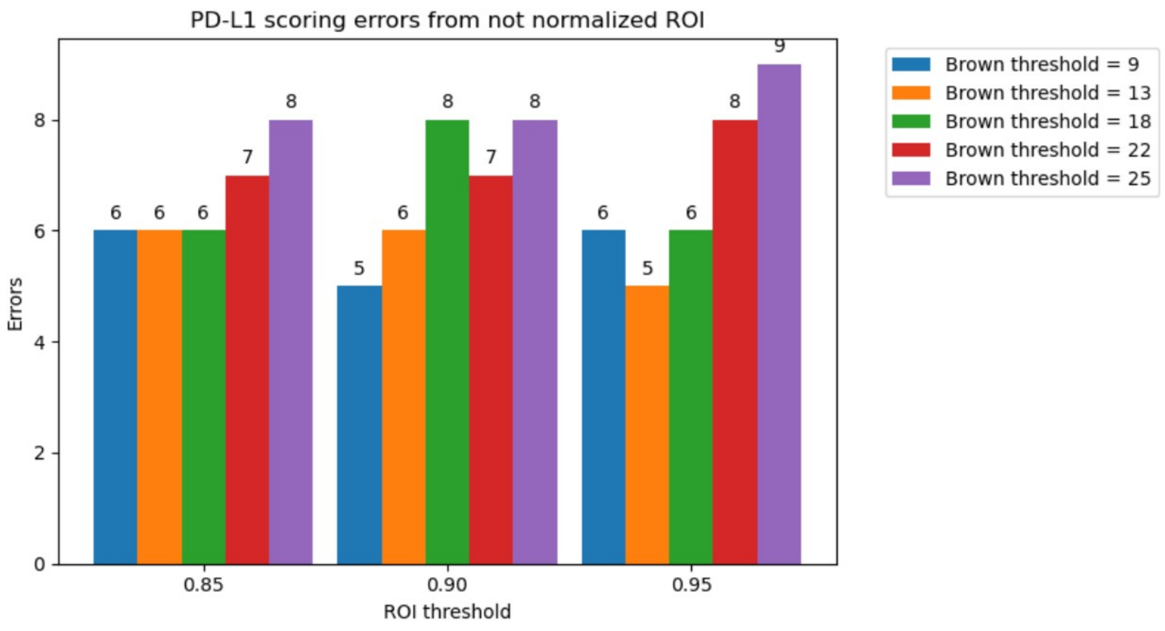


Figure 7.6: Grid search results of PD-L1 scoring base model from *not* normalized ROI map

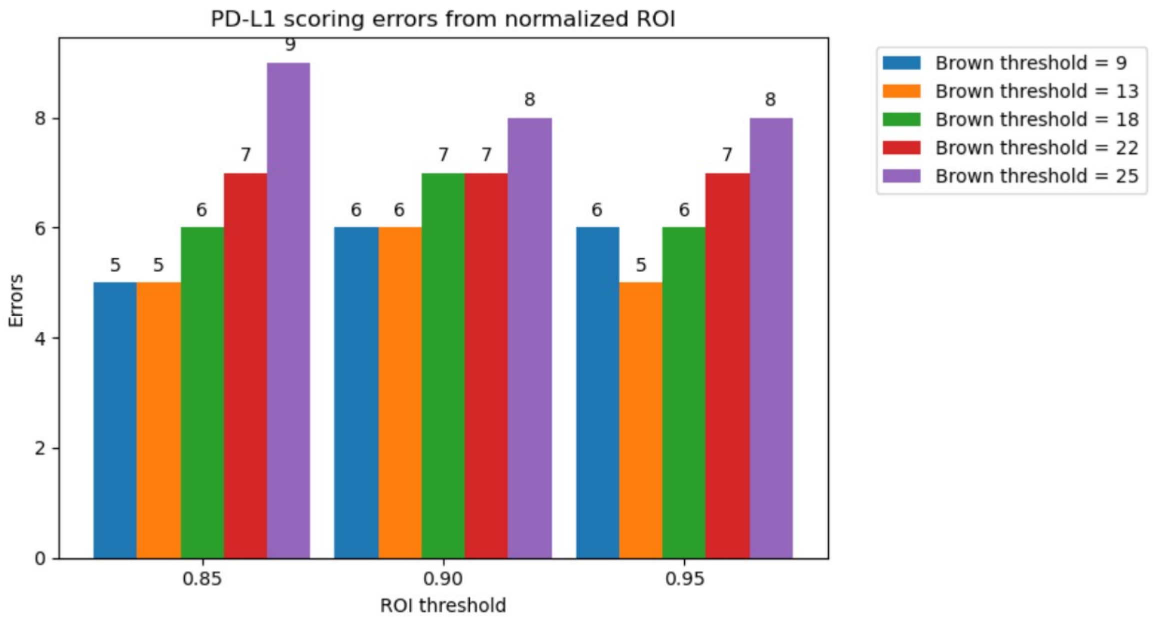


Figure 7.7: Grid search results of PD-L1 scoring base model from *normalized* ROI

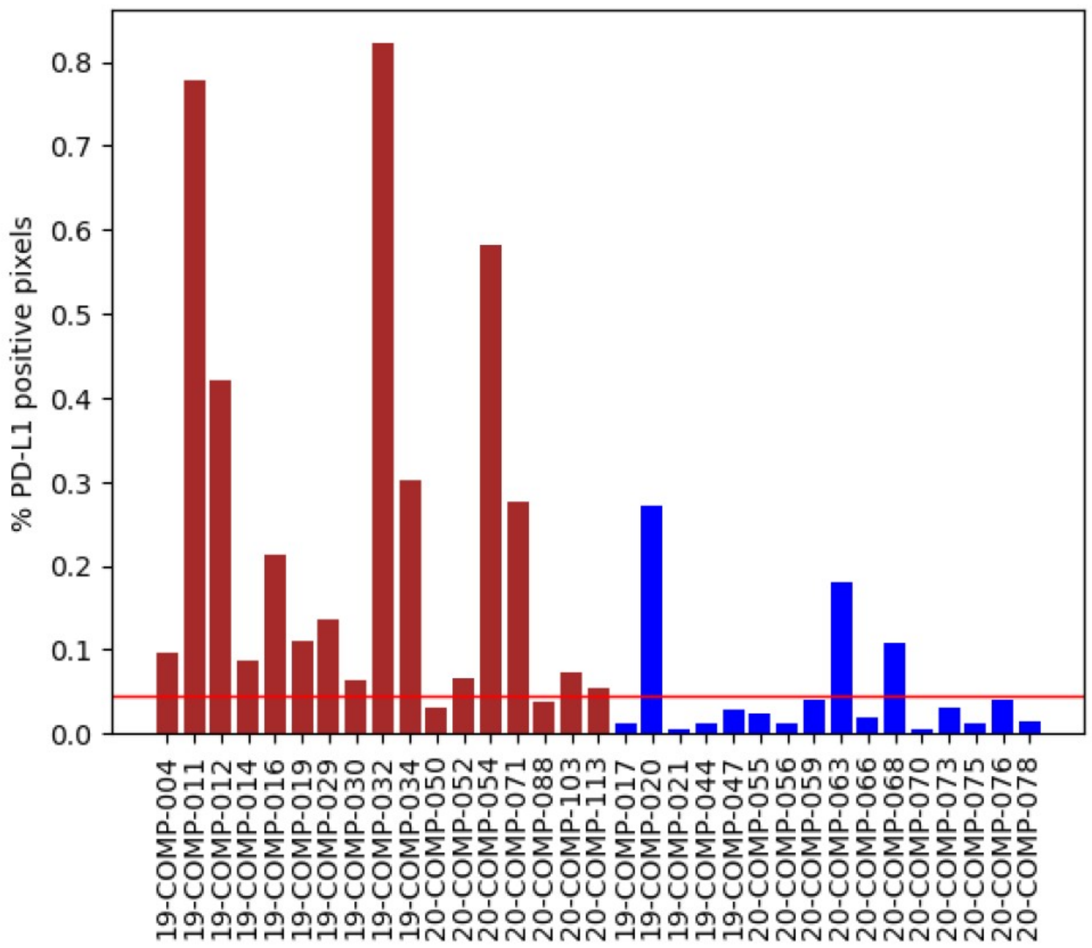


Figure 7.8: Graph representing the percentage of pixels below base_brown for each WSI in the base model with final chosen parameters. The color of the bars represents the target classification for the slide (brown for PD-L1 positive and blue for PD-L1 negative). The red line is the score_threshold.

stains for positive slides (one also has extensive artifact presence). Among the false positives, one has a very dubious light ROI, while the others present extensive brown staining artifacts.

7.4 Grid search for ML models

All 4 machine learning models, with the best hyperparameters found through the grid search, have an accuracy over 55% with cross validation. Figure 7.9 shows the accuracy of the best parameter configuration for all models. The accuracy for each model is the average of the result obtained validating on the single validation fold the model with the best found hyperparameters trained on the remaining 5 folds. Table 7.1shows the best parameters for each model.

The random forest model shows the best classification capabilities with regard of the training set with an accuracy of 87.78%, while the logistic regression the worst with 57.22%. SVM and decision tree have similar performances.

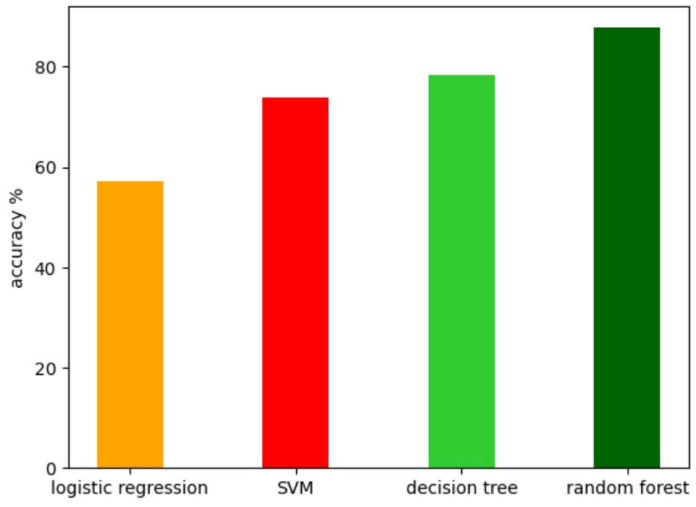


Figure 7.9: Graph representing the cross-validation accuracy of the 4 optimised ML models applied to the distribution of pixel distances from base_brown.

7.5 Test set

The test is composed of 6 WSIs (3 positive and 3 negative) and has been used to evaluate the predicting capabilities of the found models on unseen data. A comparison of the base model and ML models performances on the test set can be seen in figure

Logistic regression model best hyperparameters	
Threshold on ROI	0.95
Number of histogram bins	100
max_iter	500
C	0.1
SVM model best hyperparameters	
Threshold on ROI	0.90
Number of histogram bins	40
kernel	linear
gamma	scale
C	100
Decision tree model best hyperparameters	
Threshold on ROI	0.95
Number of histogram bins	40
Random forest best hyperparameters	
Threshold on ROI	0.95
Number of histogram bins	100
bootstrap	False
max_depth	100
max_features	log2
min_samples_leaf	1
n_estimators	100

Table 7.1: List of best hyperparameters for each ML model

7.10.

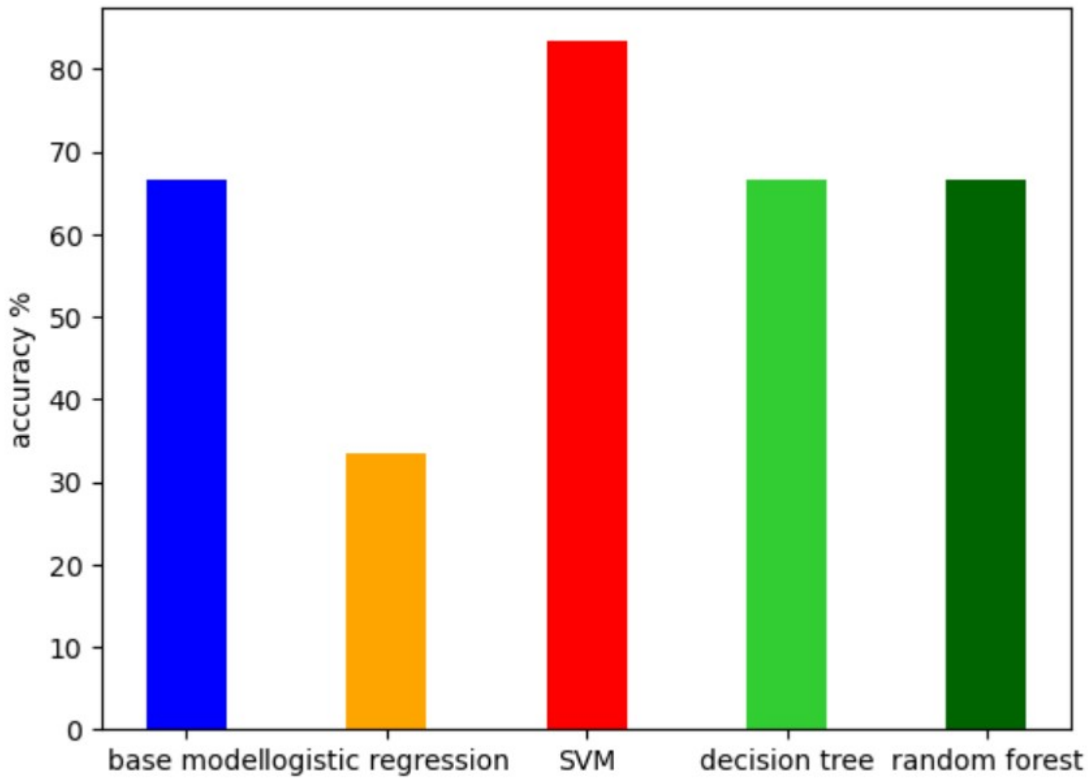
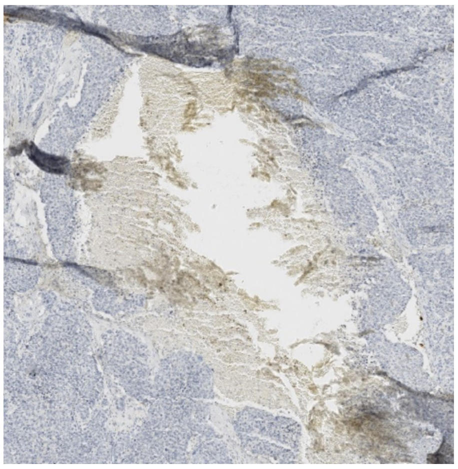


Figure 7.10: Overview of the final accuracy on the test set for the five PD-L1 positivity classifiers

7.5.1 Base model

The chosen base model with a `ROI_threshold` of 0.95 applied over the normalized float ROI map, `brown_threshold` of 9 and `score_threshold` of 0.045% was applied on the test set. Two slides out of six were misclassified; it is possible to conclude that the base model has an accuracy on the test set of 66.67%.

The two misclassified slides are both false positives, slide 20-COMP-077 and 20-COMP-065. Their PD-L1 positivity score was of 0.054% and 0.055% respectively, relatively close to the `score_threshold` of 0.045%, but enough to misclassify them. Of the 2 misclassified slides, 20-COMP-065 shows extensive areas with brown staining not reconducible to PD-L1 positivity, while 20-COMP-077 shows an high PD-L1 staining for a negative slide together with brown artifacts similar to 20-COMP-065 but less extensive (shown in Figure 7.11). These factors could have been the cause of the misclassifications because the algorithm has no means to distinguish brown artifacts (only dark, near black artifacts can be eliminated in the normalized ROI identification process). The base model shows a clear bias towards a positive classification, with a high sensitivity and a lower specificity.


(a) Examples of brown non-PD-L1 artifacts from negative 20-COMP-065

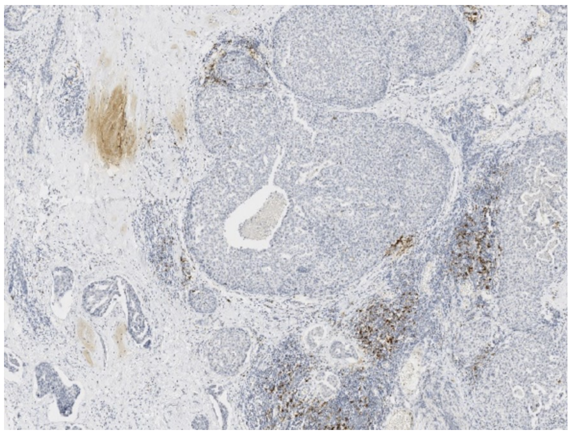

(b) Example of high PD-L1 positivity area and brown artifact for negative 20-COMP-077

Figure 7.11: Sections of misclassified false positive slides in the test set by the base model

7.5.2 Machine learning models

The results obtained by the best machine learning models on the test set are displayed in table 7.2.

Model	Accuracy	Errors	False positives	False negatives
Logistic regression	33.33%	4	1	3
SVM	83.33%	1	0	1
Decision tree	66.67%	2	2	0
Random forest	66.67%	2	2	0

Table 7.2: ML models scores on the test set

The logistic regression model is the worst of the set, confirming the low accuracy obtained with the cross-validation. The model probably does not have enough flexibility to fit our data and falls in underfitting. The SVM seems to be the most accurate model, only misclassifying the positive slide 19-COMP-026 as negative. It could be a cause of unusual brown staining on borders of a group of cells, as shown in figure 7.12.

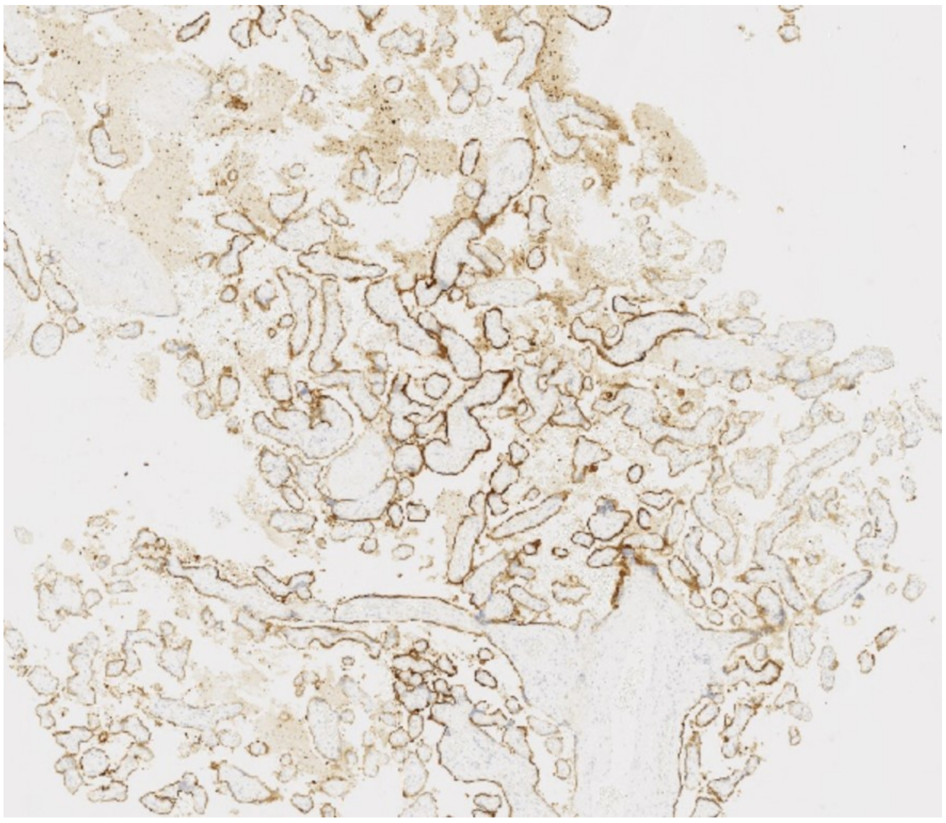


Figure 7.12: Unusual brown staining on border of grouped cells on positive slide 19-COMP-026, misclassified as negative by the SVM

Both the decision tree and the random forest allowed to obtain the same accuracy on the test set, misclassifying the same slides. The misclassified slides are the same that are erroneously classified in the test set by the base model, and the probable causes of this misclassification are reported in subsection 7.5.1. The random forest shows a drop in accuracy compared to the cross validation, which could be a possible sign of overfitting.

Chapter 8

Conclusions

The objective of this thesis was to create a model capable of classifying PD-L1 positivity on a WSI. This accomplished with a sufficient degree of accuracy, having consistent results for the multiple models analyzed. Various models showed a good accuracy, but in particular the SVM in regard to the ML model pipeline for PD-L1 scoring seemed the most promising for this task. Still, our results are not as accurate as a state of the art PD-L1 estimation model as the one in section 2.3.2, which accuracy is well above 90%, but with substantial differences on the richness of the dataset.

The heuristic model for the tumor ROI also seemed to generate a rather accurate estimation, even though a systematic estimate on its accuracy could not be made as annotations regarding the ROI were not available.

Although satisfactory results were achieved, we cannot make strong assumptions on the generalization capabilities of the models, as the datasets on which the model was constructed and tested, in spite of having some degree of variance, were very limited in size. This is even more evident considering the vast biological and morphological variability of cancer, as well as the technical variability introduced by the staining and preparation processes in histopathology[16].

The inherent modular nature of the models, which divide ROI identification and PD-L1 scoring, grants more flexibility for future improvements to the models pipeline. For example, more advanced approaches could be explored only in regard to ROI identification, or other ML models could be added in the final part of PD-L1 scoring.

These possible future advancements are largely subordinate to the presence of new data, both in the form of more WSIs or detailed annotations. Data scarcity (both in sense of quantity and of labeling provided for each example) was the greatest limitation for the model development and selection. Full machine learning approaches (possibly using deep learning) could be pursued with extensively annotated slides as shown in subsection 2.3.2, or with a large amount of slides to train on as seen in subsection 2.3.3.

These improvements, as explained before, could be performed also only for a part of the modelling pipeline; for example, the introduction of high-level annotations of the ROI (instead of time consuming cell-level annotations) could be sufficient to pursue a more accurate full machine learning approach for ROI identification.

Bibliography

[1] Anaconda. URL https://www.anaconda.com/.

[2] Camelyon17 challenge. URL https://camelyon17.grand-challenge.org/.

[3] Project jupyter. URL https://jupyter.org/.

[4] Matplotlib. URL https://matplotlib.org/.

[5] Numpy. URL https://numpy.org/.

[6] Opencv, . URL https://opencv.org/.

[7] Openslide, . URL https://openslide.org/.

[8] 1.10. decision trees, . URL https://scikit-learn.org/stable/modules/tree.html.

[9] Scikit-image, . URL https://scikit-image.org/.

[10] Scikit-learn, . URL https://scikit-learn.org/.

[11] Logistic regression, . URL https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html.

[12] Random forest classifier, . URL https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.

[13] Ventana pd-l1 (sp142) assay, interpretation guide for triple-negative breast carcinoma (tnbc), 2019.

[14] Famke Aeffner, Markd Zarella, Nathan Buchbinder, Marilynm Bui, Matthewr Goodman, Douglasj Hartman, Giovannim Lujan, Mariama Molani, Anilv Parwani, Kate Lillard, and et al. Introduction to digital image analysis in whole-slide imaging: A white paper from the digital pathology association. *Journal of Pathology Informatics*, 10(1):9, Mar 2019. doi: 10.4103/jpi.jpi_82_18.

[15] Patel Janakkumar Baldevbhai and R. S. Anand. Color image segmentation for medical images using l*a*b* color space. *IOSR Journal of Electronics and Communication Engineering*, 1(2):24–45, 2012. doi: 10.9790/2834-0122445. URL https://www.iosrjournals.org/iosr-jece/papers/vol1-issue2/M0122445.pdf?id=4572.

[16] Gabriele Campanella, Matthew G. Hanna, Luke Geneslaw, Allen Miraflor, Vitor Werneck Krauss Silva, Klaus J. Busam, Edi Brogi, Victor E. Reuter, David S. Klimstra, Thomas J. Fuchs, and et al. Clinical-grade computational pathology using weakly supervised deep learning on whole slide images. *Nature Medicine*, 25(8):1301–1309, 2019. doi: 10.1038/s41591-019-0508-1. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7418463/.

[17] Trevor Hastie, Jerome Friedman, and Robert Tisbshirani. *The elements of Statistical Learning: Data Mining, Inference, and prediction.* Springer, 2017.

[18] Simon Haykin. *The Neural Networks a comprehensive foundation.* Macmillan College Publishing Company.

[19] Sanghun Lee, Joonyoung Cho, and Sun Woo Kim. Automatic classification on patient-level breast cancer metastases. *Camelyon-17*, Feb 2019. URL https://grand-challenge-public-prod.s3.amazonaws.com/evaluation-supplementary/80/46fc579c-51f0-40c4-bd1a-7c28e8033f33/Camelyon17_.pdf.

[20] Lidija Mandic, Sonja Grgic, and Mislav Grgic. Comparison of color difference equations. In *Proceedings ELMAR 2006*, pages 107–110, 2006. doi: 10.1109/ELMAR.2006.329526. URL https://ieeexplore.ieee.org/document/4127499.

[21] Khairul Anuar Mat Said, Asral Jambek, and Nasri Sulaiman. A study of image processing using morphological opening and closing processes. *International Journal of Control Theory and Applications*, 9:15–21, 01 2016. URL https://www.iosrjournals.org/iosr-jece/papers/vol1-issue2/M0122445.pdf?id=4572.

[22] Tom M. Mitchell. *Machine learning.* McGraw Hill, 1997.

[23] Liron Pantanowitz, Navid Farahani, and Anil Parwani. Whole slide imaging in pathology: Advantages, limitations, and emerging per-

spectives. *Pathology and Laboratory Medicine International*, page 23, 2015. doi: 10.2147/plmi.s59826. URL https://www.dovepress.com/whole-slide-imaging-in-pathology-advantages-limitations-and-emerging-p-peer-re

[24] Florian Schütz, Stefan Stefanovic, Luisa Mayer, Alexandra von Au, Christoph Domschke, and Christof Sohn. Pd-1/pd-l1 pathway in breast cancer. *Oncology Research and Treatment*, 40(5):294–297, 2017. doi: 10.1159/000464353. URL https://pubmed.ncbi.nlm.nih.gov/28346916/.

[25] J. Wu, C. Liu, X. Liu, W. Sun, L. Li, Y. Zhang, J. Zhang, H. Wang, X. Liu, X. Yang, and et al. Deep learning approach for automated cancer detection and tumor proportion score estimation of pd-l1 expression in lung adenocarcinoma. 2020. doi: 10.1101/2020.05.31.126797. URL https://www.biorxiv.org/content/10.1101/2020.05.31.126797v1.full.pdf.