

BAL IMAGES ANALYSIS FOR THEIR AUTOMATIC QUANTIFICATION

A Degree Thesis

Submitted to the Faculty of the
Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona
Universitat Politècnica de Catalunya

by

Alejandro Campos Almazán

In partial fulfilment
of the requirements for the degree in
Enginyeria de Tecnologies I Serveix de Telecomunicació – Sistemes Audivisuals

ENGINEERING

Advisors: Ferran Marqués Acosta, Montse Pardás Feliu

Barcelona 21/06/2021

Final Report

Abstract

Bronchoalveolar lavage (BAL) images are a medical test from which different pathologies can be extracted based on their cellular distribution. In the hospital La Vall d'Hebron technicians make manual count to determine this cell distribution. We did not have any labelled BAL image, because this was the first contact with the problem. In this project, maxtree-based solution is used to do the first segmentation, and then, a correcting process is carried out in order to correctly label 56 BAL images. With this small dataset, and taking into account the maxtree results in some aspects were not as good as we expected, we decided to test and train a CNN based in the U-Net. We applied specific techniques for small datasets and we tested different parametrizations of the network. Finally, we obtain 70% IoU global score approx. in the validation of the CNN.

Acknowledgements

Carrying out this project has been a great experience for me, full of small successes and failures, of motivation and frustration at the same time, of better and worse moments and ideas. But especially, it was an experience full of learning, which result is not as important to me as everything I have learned along the way.

I would like to express my deep gratitude to professor Ferran Marques and professor Montse Pardàs, my research supervisors, for trust me to carry out this project, their constancy, their patience guidance, their valuable and constructive suggestions, their support, encouragement and their great knowledge and experience. Without them, none of this would have been possible.

Also; I want to thank Philippe Salembier for giving me the opportunity to work with the Maxtree software, the cornerstone of my project and for solving all my problems and doubts with extreme patience; Rocco Clemente Bonjour, for showing, providing and explaining me his code to generate tiles from WSI; Adrià Marcos Morales, whose help was essential to correct the output segmentations of the maxtree algorithm, thank you for being so patient and having spent so much time on my problem.

And finally, I would like to express my very great appreciation to Ignasi Nogueiras, who helped me in the worst moments, with his best advices, his wisdom, his big patience, support and his unconditional friendship. For all this, my sincerest gratitude my friend.

Revision history and approval record

Revision	Date	Purpose
0	09/03/2021	Document creation
1	24/06/2021	Document revision

DOCUMENT DISTRIBUTION LIST

Name	e-mail
Alejandro Campos Almazán	alejandro.campos.almazan@estudiantat.upc.edu
Ferran Marqués	ferran.marques@upc.edu
Montse Pardàs	montse.pardas@upc.edu

Written by:		Reviewed and approved by:	
Date	21/06/2021	Date	24/06/2021
Name	Alejandro Campos A.	Name	Ferran Marques
Position	Project Author	Position	Project Supervisor

Table of contents

Abstract	0
Acknowledgements	1
Revision history and approval record	2
Table of contents	3
List of Figures	5
List of Tables:	5
1. Introduction	8
1.1. Introduction to the problematic	8
1.2. Introduction to the data	11
2. State of the art of the technology used or applied in this thesis:	12
2.1. BAL Images Analysis	12
2.2. Semantic Image Segmentation	12
2.3. Maxtree	12
2.4. Convolutional Neural Network (CNN)	14
2.4.1. Unet	14
3. Project Development	16
3.1. Appropriate Image Selection	16
3.2. Maxtree-based Solution	18
3.2.1. Image Simplification.....	18
3.2.2. Macrophage Detection.....	21
3.2.3. Lymphocytes, Polymorphonuclear and Eosinophiles Detection	31
3.2.4. Mask creation.....	38
3.2.5. Manual Labelling Corrections	39
3.2.6. VGG Image Annotator	39
3.3. Convolutional Neural Network, U-Net	42
3.3.1. Introduction.....	42
3.3.2. Network Architecture	42
3.3.3. Data Augmentation	42
3.3.4. Transfer Learning.....	43
3.3.5. Training mini-batch size.....	43
3.3.6. Activation Function	43
3.3.7. Classification.....	44
3.3.8. Adam Optimizer	45
3.3.9. Dice Loss Function	45
So, the generic formula to compute it. [17]	46
4. Results	47

4.1. Metrics, IoU	47
4.2. Final Results.....	47
4.2.1. Loss Function	49
4.2.2. Optimizer	49
4.2.3. Learning Rate Annealing	50
4.2.4. Number of epochs	51
4.2.5. Global and per Class IoU Score	52
4.2.6. Pixel detection IoU score.....	53
4.2.7. Tricky solution.....	53
4.2.8. Maxtree Results.....	54
5. Conclusions and future development.....	55
5.1. Conclusions	55
5.2. Future Development	55
5.2.1. New techniques to classify lymphocytes.....	55
5.2.2. Improve Transfer Learning.....	55
5.2.3. Improve Class Imbalance	56
5.2.4. Squamous cellularity detection.....	56
6. Bibliography.....	57

List of Figures

Figure 1: Bronchoalveolar Lavage (BAL) Image Cut.....	8
Figure 2: Macrophages	9
Figure 3: Lymphocytes with a macrophage.....	9
Figure 4: Polymorphonuclears with a macrophage	9
Figure 5: Eosinophiles	9
Figure 6: Bronchial cilindric cell.....	10
Figure 7: Squamous mouth cell.....	10
Figure 8: Blood & Mucus	10
Figure 9: Semantic Segmentation of a street picture [3]	12
Figure 10: Tree representation of BAL Image cut (inverse)	13
Figure 11: U-net architecture. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations. [9].....	15
Figure 12: Tile suitable for analyse vs not suitable	16
Figure 13: Normal distribution of the two tile types	17
Figure 14: Spurious nucleolus.....	18
Figure 15: Spurious maxima located in mucus	18
Figure 16: Opening vs Reconstruction	19
Figure 17: Closing vs Dual Reconstruction	19
Figure 18: Granulometry of one image.....	20
Figure 19: Closing vs Dual Reconstruction	20
Figure 20: Top Hat 3 vs 4 pixels radius structuring element	20
Figure 21: Top Hat 7 vs 8 pixels radius structuring element	21
Figure 22: Reconstruction with 3 pixels radius structuring element	21
Figure 23: Maxtree Representation.....	22
Figure 24: Macrophage cytoplasm analysis	23
Figure 25: Macrophage Soup.....	24
Figure 26: Macrophage nucleus analysis.....	25
Figure 27: Macrophage Detection Example	26
Figure 28: Macrophage cytoplasm image of candidates.....	26
Figure 29: Macrophage cytoplasm opening and dual reconstruction.....	27
Figure 30: Macrophage nucleus hole filled.....	28
Figure 31: Macrophage nucleus cleaned	28

Figure 32: Macrophage image of candidates cleaning example 1	28
Figure 33: Macrophage image of candidates cleaning example 2	29
Figure 34: Macrophage detection example 1	30
Figure 35: Macrophage detection example 2	30
Figure 36: Polymorphonuclear / Eosinophile / Lymphocyte	31
Figure 37: Lymphocytes, polymorphonuclears and eosinophiles detection	32
Figure 38: Nuclei Overlapped.....	33
Figure 39: Example of how 4 pixels radius structuring element remove relevant data	34
Figure 40: Second cut example.....	34
Figure 41: Distance Function	35
Figure 42: Analysis by maxima of distance function 1	36
Figure 43: Analysis by maxima of distance function 2	36
Figure 44: Example of small cells detection 1	37
Figure 45: Example of small cells detection 2	38
Figure 46: Output masks examples.....	38
Figure 47: Correcting masks in the VGG Annotator.....	39
Figure 48: Images Segmentation Procedure.....	40
Figure 49: Correction example 2	41
Figure 50: Correction example 1	41
Figure 51: Data Augmentation in play [11]	42
Figure 52: ReLU vs Sigmoid [26]	43
Figure 53: Sigmoid vs ReLU	44
Figure 54: Output segmentation maps	45
Figure 55: Sørensen–Dice coefficient [25]	45
Figure 56: IoU [19].....	47
Figure 57: Validation Results	48
Figure 58: Adam vs Adamax	50
Figure 59: Static learning rate vs variable	51
Figure 60: IoU Score along 180 epochs.....	51
Figure 61: Correlation between class representation and its IoU.....	53
Figure 62: Lymphocyte vs Polymorphonuclear	55

List of Tables:

Table 1: Mean grey level of suitable images for analysis.....	17
Table 2: Mean grey level of images not suitable for analysis.....	17
Table 3: Maxtree nodes applying different morphological techniques	19
Table 4: Parameters final values.....	24
Table 5: Macrophage nuclei parameters final values.....	25
Table 6: Small cells cytoplasm parameters final values.....	31
Table 7: Small cells nuclei parameters final value	32
Table 8: Results Table	48
Table 9: Loss Function results comparison.....	49
Table 10: Correlation between class representation and its IoU.....	52
Table 11: CNN vs Maxtree	54

1. Introduction

1.1. Introduction to the problematic

Bronchoalveolar lavage (BAL) is a diagnostic method of the lower respiratory system, in which a bronchoscope is passed through the mouth or nose into an appropriate airway in the lungs, with a measured amount of fluid introduced and then collected for examination. After fluid collection, samples are centrifuged to distribute quite homogeneously its content and stained to clearly differentiate its cells, in our case, we work with Papanicolaou stain. This method is typically performed to diagnose pathogenic infections of the lower respiratory airways (leading to, for example pneumonia, COVID-19, etc.) [1]

Once the fluid is collected, centrifuged and stained, it is ready for the inspection. It can be observed directly in a microscope or scanned to produce whole slide images (WSI), which can be digitally stored and managed.

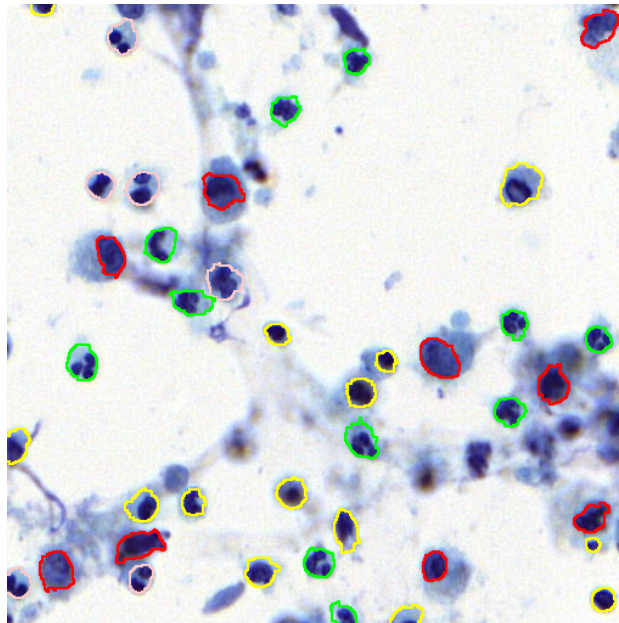


Figure 1: Bronchoalveolar Lavage (BAL) Image Cut

In these whole slide images of BAL, we can find commonly two groups of cells.

The first group, contains **inflammatory cells**, which can determine the kind of inflammation present in the sampled. They are the main cells in BAL test, commonly they represent 95% of the cells in the sample, being the main object of study. In this group we can find:

1. **Macrophages: (red contours):** big cells with big nucleus, they contain lot of cytoplasm.

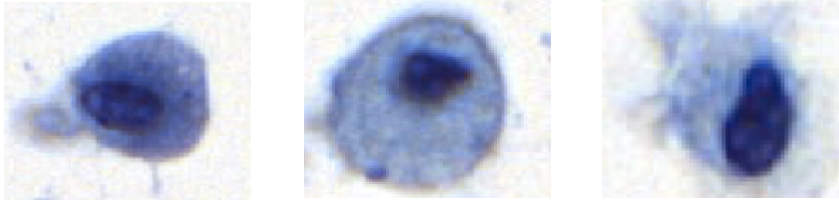


Figure 2: Macrophages

2. **Lymphocytes (yellow contours)**: small cells, round and dark nucleus; they have very little cytoplasm, we almost only see the nucleus.

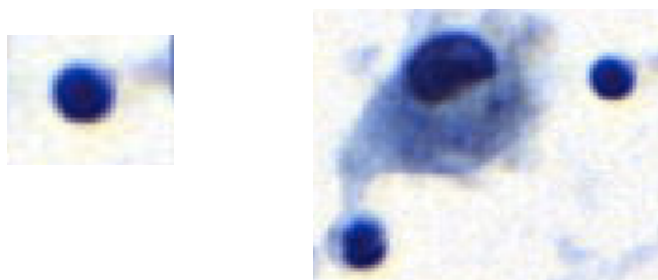


Figure 3: Lymphocytes with a macrophage

3. **Polymorphonuclears / neutrophils (green contours)**: small as lymphocytes, but they do not have one nucleus, they have 3 or 4.

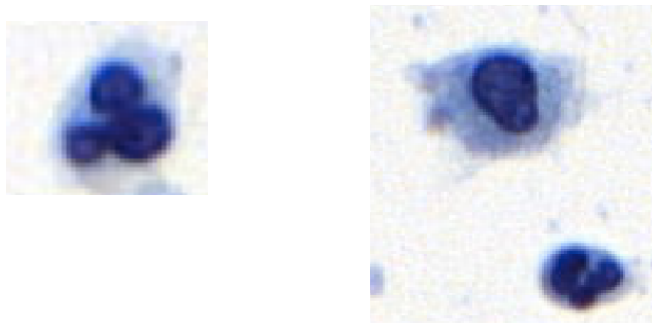


Figure 4: Polymorphonuclears with a macrophage

4. **Eosinophiles (pink contours)**: very similar to polymorphonuclears, but they do not have 3 or 4 nuclei, they only have 2.



Figure 5: Eosinophiles

Doctors can estimate the different kinds of inflammation depending on the appearance frequency of inflammatory cells types

1. **80 – 90 % of macrophages:** no inflammation, normal sample.
2. **+ 20 % of lymphocytes:** chronic inflammation (tuberculosis).
3. **+ 20 % of polymorphonuclears:** sharp inflammation (pneumonia, presence of bacteria).

The second group of cells we can find in these samples are **mucous cells**, which in our case are treated as contamination, provided that they do not exceed the 5% of the total cells. There are different types of mucous cells in BAL samples:

1. **Bronchial cellularity:** cells coming from bronchus



Figure 6: Bronchial cilindric cell

2. **Squamous cellularity:** cells coming from mouth

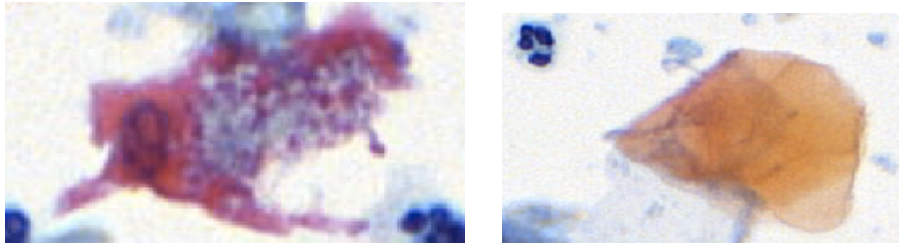


Figure 7: Squamous mouth cell

3. **Mucus & Blood**

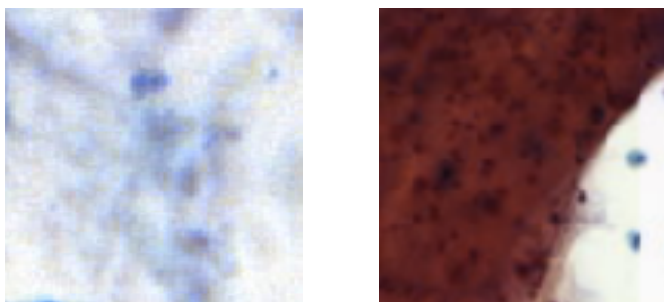


Figure 8: Blood & Mucus

1.2. Introduction to the data

Images coming out of the scanner are whole slide images (WSI), also known as virtual microscopy, refers to scanning a complete microscope slide and creating a single high-resolution digital file. This is commonly achieved by capturing many small high-resolution images and then montaging them to create a full image of a histological section [2]. These WSI are big to manage with them, that is why we break them up into smaller pieces many, that we call tiles. They are 512 pixels wide by 512 pixels high, with 30 overlapping pixels. They have enough resolution to differentiate cells and enough cells to make a representative counting,

The main goal of this project would be the cell quantification in WSI images, because it is clinically relevant. This quantification can be carried out by counting the connected components in images of semantic segmentation of the inflammatory cell types.

Taking into account the state of the art, the best solution to implement semantic segmentation are convolutional networks, but they require having a ground truth which we do not have. So, we propose to develop a system based on classical techniques of image processing, in order to have the first approximation to the correct segmentation and ease manual labelling to generate this ground truth and extract useful information for future development.

2. State of the art of the technology used or applied in this thesis:

2.1. BAL Images Analysis

According to La Vall d'Hebron doctors, the classical method to analyse these images is not automatic. Images are observed by technicians, which count randomly 100 or 200 of cells, and with this small sample they calculate their frequency distribution. They can do that because, given a WSI, its statistic is quite homogeneous in all its tiles, due to the centrifugation step. After the manual counting, technicians highlight relevant areas in order to help doctors to analyse inflammatory cells in images.

2.2. Semantic Image Segmentation

Image segmentation is a computer vision task in which we label specific regions of an image according to what is being shown. The goal of semantic image segmentation is to label each pixel of an image with a corresponding class of what is being represented.

One important thing to note is that instances of the same class are not separated; we only care about the category of each pixel. In other words, if you have two objects of the same category in your input image, the segmentation map does not inherently distinguish these as separate objects [3].

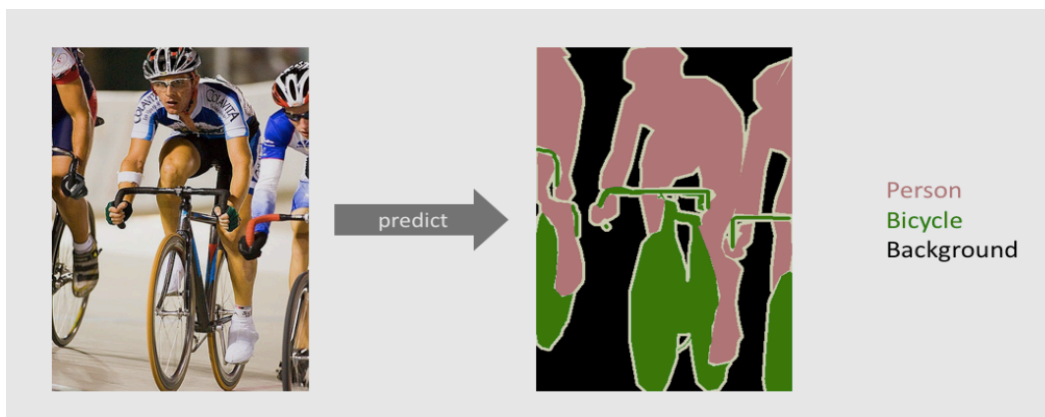


Figure 9: Semantic Segmentation of a street picture [3]

2.3. Maxtree

Maxtree and Mintree are image (or signal) representations created by structuring as a tree the connected components resulting from threshold decomposition. It provides a multiscale description of extremas of images and signals and are, among other things, one of the classical ways to build connected operators. These trees can also be populated with attributes, resulting in Graph attribute signals that can themselves be processed [4].

A Maxtree describes the entire set of connected components resulting from the threshold decomposition of upper-level sets. The resulting connected components are ordered by inclusion and structured in a tree. The tree leaves represent the image maxima and the root node the entire image support. A Maxtree can then be viewed as a multiscale description of the image maxima [5].

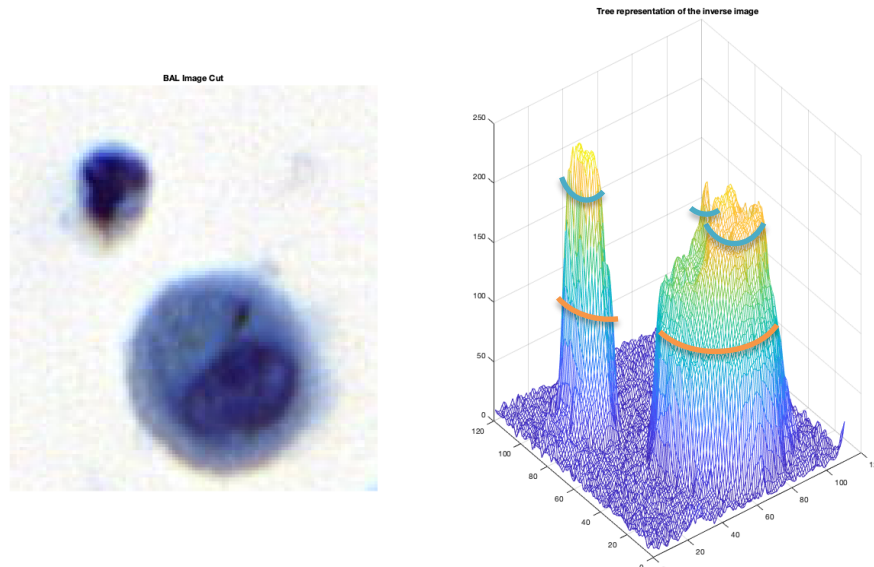


Figure 10: Tree representation of BAL Image cut (inverse)

Maxtree processing toolbox allows easy creation and processing of attributes in Maxtree or Mintree. These attributes defined on the tree structure can be considered as graph signals and processed with the corresponding graph signal processing tools. With this toolbox we can make a distinction between the pixels that are associated to a node and the connected components created by the threshold decomposition.

The nodes in a Maxtree actually represent a structuring by inclusion of the connected components created by threshold decomposition. However, the set of pixels defining the connected component associated to a node can be obtained as the union of the pixels stored in a node and all its descendant nodes [4].

The features used to develop the project are the following [4]:

- **Area:** Compute the number of pixels of the connected component associated to a current node. This number is equal to the sum of the number of pixels stored in the current node and all its descendant nodes.
- **MeanGrayLevel:** Mean value of the GrayLevel field in the connected component (that is the area associated to the current node and all its descendant node).
- **MaxLeafValue:** Compute the height of the largest branch of a node.

More features are currently available, but after some tests, they were rejected for our problematic. Also, trees can be used to generate images, filtering the different nodes based on their attributes. This is the restitution step.

Maxtree is the classic image processing tool we will use to have a first approximation to the semantic segmentation.

2.4. Convolutional Neural Network (CNN)

Convolutional neural networks are a type of deep artificial neural networks widely used in the field of computer vision. They have been applied to many tasks, including image classification, super-resolution and semantic segmentation [6].

The role of the CNN is to reduce the images into a form which is easier to process, without losing features which are critical for getting a good prediction [7].

Convolutional Neural Network are currently the state of art technique in image segmentation field of research. Since astounding results on ImageNet has been demonstrated, all other methods have rapidly been abandoned [8].

2.4.1. Unet

U-Net architecture consists of a contracting path (left side), a fully connected path and an expansive path (right side). In total the network has 23 convolutional layers.

The contracting path follows the typical architecture of a convolutional network. It consists of the repeated application of two 3x3 convolutions (unpadded convolutions), each followed by an activation function and a 2x2 max pooling operation with stride 2 for downsampling. At each downsampling step we double the number of feature channels.

Every step in the expansive path consists of an upsampling of the feature map followed by a 2x2 convolution (“up-convolution”) that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, each followed by an activation function. At the final layer a 1x1 convolution is used to map each 64-component feature vector to the desired number of classes.

To allow a seamless tiling of the output segmentation map, it is important to select the input tile size such that all 2x2 max-pooling operations are applied to a layer with an even x- and y-size [9].

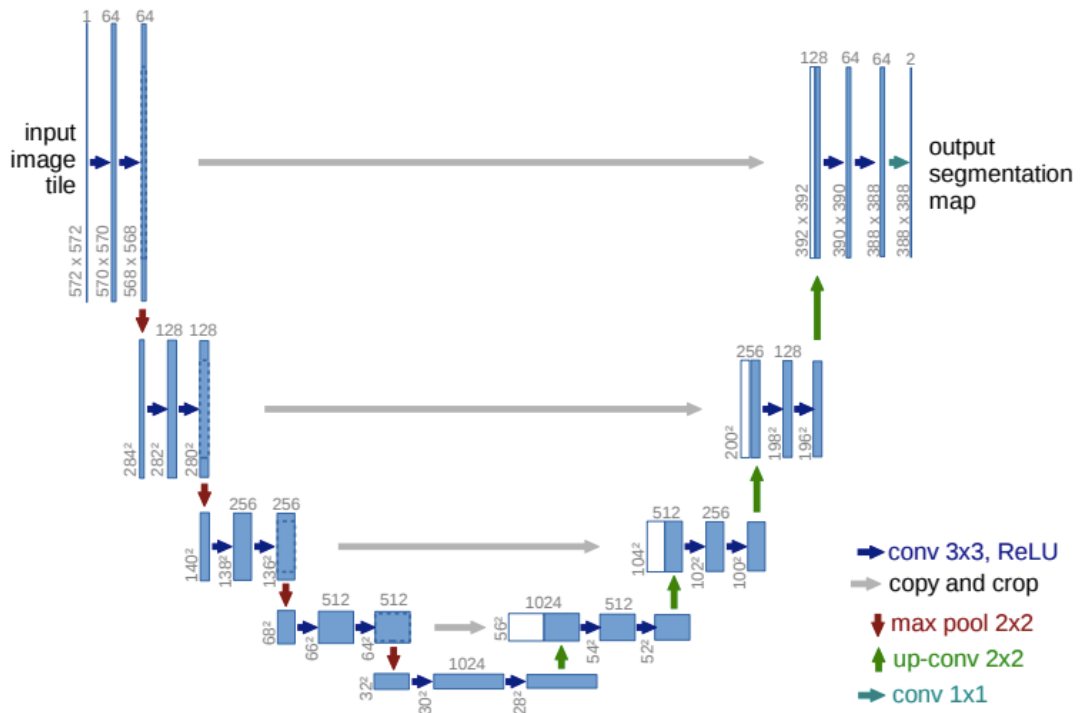


Figure 11: U-net architecture. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations. [9]

3. Project Development

As we said in the state of the art, a CNN trained on a large and versatile dataset, learns information about object characteristics that is generic enough to transfer to objects that are not in the original dataset.

But what happen if we do not have that annotated dataset? We have just received the BAL images from la Vall d'Hebron hospital, we do not have any labelled image. So, as first approximation, we decided to use classical techniques to tackle the problem. Because the manual labelling is a hard and tedious task, having a first technique that classifies quite well the pixels of the images into the different classes, would be a good starting point, which would reduce the manual labelling task.

Having studied different classical segmentation techniques for our problem, we decided to implement a Maxtree solution on grayscale images. Due to its tree and branch structure and its good handling of connected components we thought that the Maxtree solution would be suitable for our problem.

3.1. Appropriate Image Selection

The first step, is to decide which tiles are suitable for segmentation and which are not. Because some tiles are clean and they have cells that can be well differentiate, but others can have much contamination, and classify cells can be a difficult task, also for doctors. For that reason, in order to carry out the correct and accurate count of the inflammatory cell types, it is necessary to determine which tiles of the WSI are suitable for a correct classification. Dismiss not suitable tiles would not be a problem from a statistical point of view, taking into account that WSI statistics are quite homogeneous in every tile due to the centrifugation step and that a WSI can contain hundreds of thousands of cells. Also, when technicians analyse these images in the hospital, they ignore the parts with a lot of mucus, where the counting of inflammatory cells is very difficult, so in some way, they do the same.

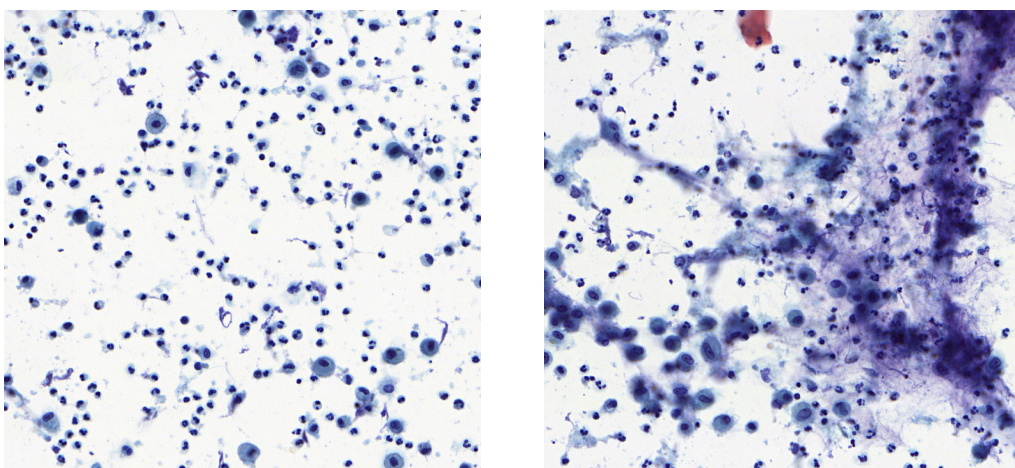


Figure 12: Tile suitable for analyse vs not suitable

To carry out this task, a thresholding algorithm is used, based on the average grey level of the image. If the mean grey level of the image is lower than a threshold, the tile is automatically rejected.

To determine the threshold, we analyse some images, and here we have an example of the mean grey level of ten images suitable for the analysis and ten images which are not.

Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	Image 7	Image 8	Image 9	Image 10
223	225	219	210	232	220	222	216	236	228

Table 1: Mean grey level of suitable images for analysis

Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	Image 7	Image 8	Image 9	Image 10
193	200	196	191	198	213	208	202	212	208

Table 2: Mean grey level of images not suitable for analysis

In the next figure we can see the probability density function of both tile types, approximating them as gaussian and computing mean and variance about twenty images of each class.

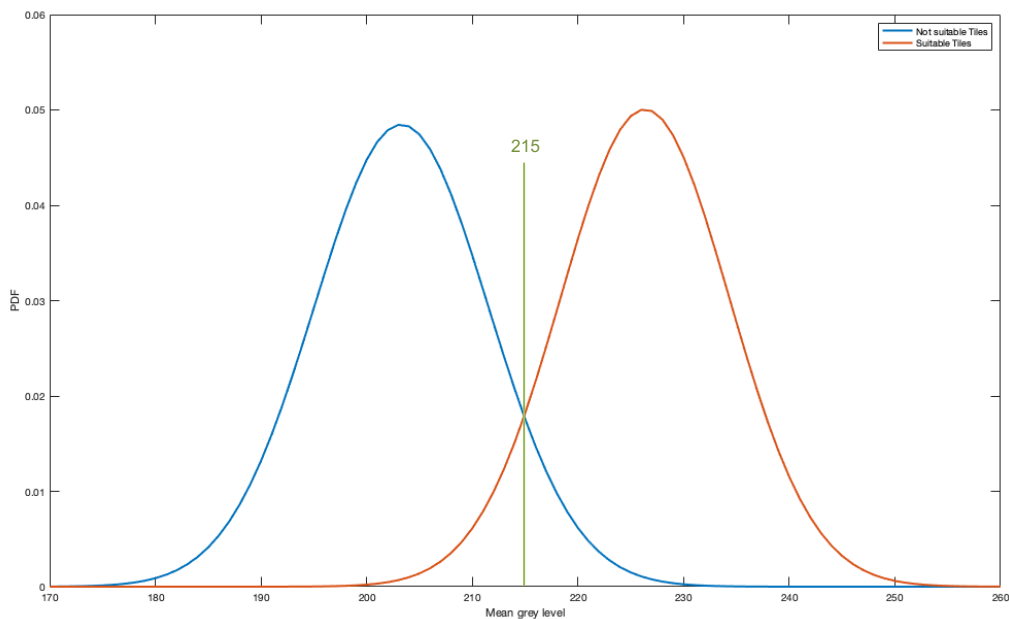


Figure 13: Normal distribution of the two tile types

As we can see, 215 is the mean grey level where the curves intersect, so it is where we decide to put the threshold.

3.2. Maxtree-based Solution

3.2.1. Image Simplification

As we said before, we are going to use Maxtree algorithms to develop the first segmentation. But before computing it, reducing the number of nodes of the tree and remove spurious candidates without losing relevant image data would be important in order to reduce the computation time and to not have spurious detection. For that reason, morphological cleaning techniques have been used; opening, closing, reconstruction with the original eroded image as marker and dual reconstruction with the original dilated image as marker.

Best techniques should reduce the maximum number of nodes without losing relevant data, the nuclei, their shape and their separation, so it is important to not remove nuclei and to not increase the overlap between them, either.

Opening and opening by reconstruction of erosion are two morphological techniques which remove those maxima smaller than the structuring element, so the goal of applying them is to remove small maximums that bother us or do not contribute to us when analysing the tree. For example; spurious maxima located in the mucus, nucleolus inside cells, or variations in the grey level of the nuclei. In Figure 15: Spurious maxima located in mucus and Figure 14: Spurious nucleolus we can see how applying morphological techniques can help reducing remove these unwanted maxima.

Closing and closing by reconstruction of dilation are two morphological techniques which remove those minima smaller than the structuring element, so the goal of applying them is to remove small minima which do not contribute to us when analysing the tree without remove relevant data.

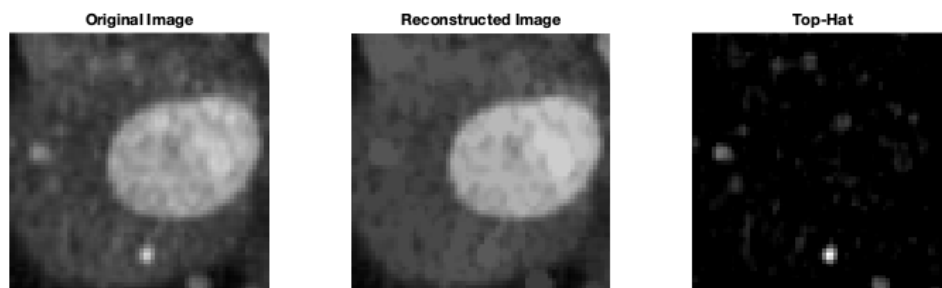


Figure 14: Spurious nucleolus

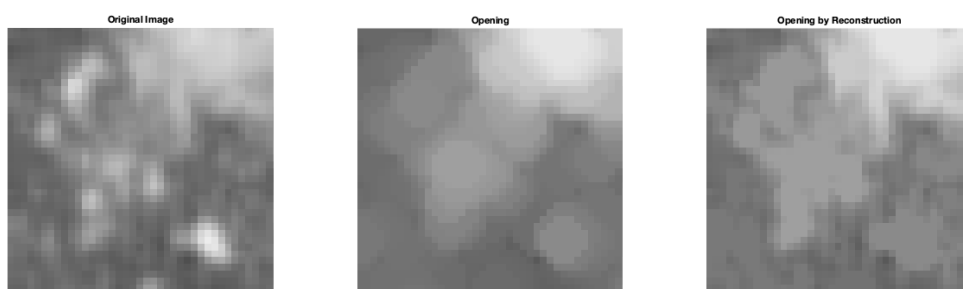


Figure 15: Spurious maxima located in mucus

It is important to remark that in order to have more visual information, images are inverted in grey level scale, to look for maxima and not for minima. This is applied in all the project, so when we talk about maxima, are actually minima of the original image and vice versa.

3.2.1.1. Opening vs opening by reconstruction of erosion and closing vs closing by reconstruction of dilation

Firstly, we are going to test which operation most reduce the number of nodes in the tree. In the next table we can see the average number of nodes that different images Maxtrees have depending on the cleaning technique we apply. These averages are computed with 43 different images, with 3 radius pixels disk structuring element.

	Image	Opening	Opening by reconstruction	Closing	Closing by dual reconstruction
Average N° Nodes	36.816	7.754	6.664	13.544	36.531

Table 3: Maxtree nodes applying different morphological techniques

Opening by reconstruction of erosion and closing are the two operations which reduce the largest number of nodes. But it is also very important to preserve the relevant data for the correct segmentation of the image. For that reason, opening by reconstruction of erosion and closing by reconstruction of dilation are applied. These are the techniques who best preserve the natural shape of the cells and do not cause overlap between cells.

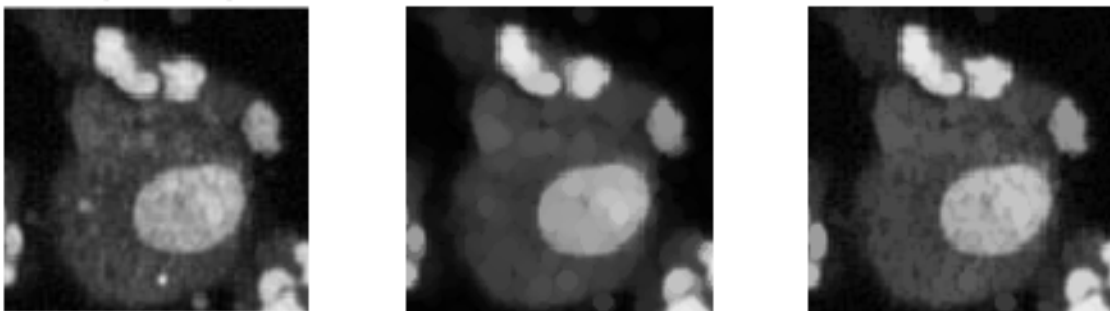


Figure 16: Opening vs Reconstruction

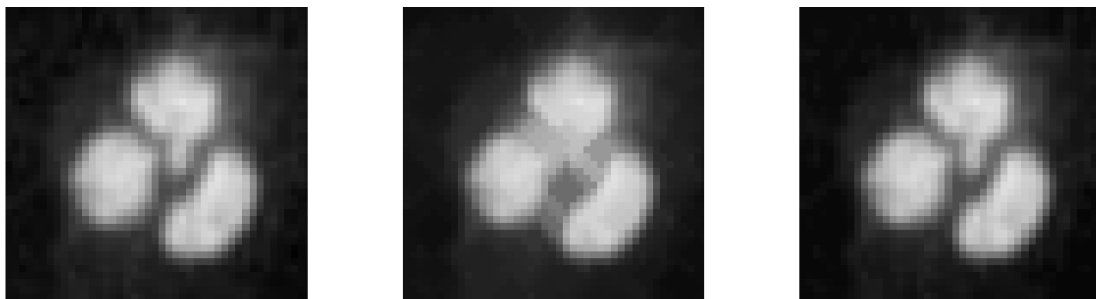


Figure 17: Closing vs Dual Reconstruction

3.2.1.2. Size of structuring element

In order to estimate the size of the elements in the image and which radius size of the structuring element starts to remove relevant data, granulometry techniques have been applied. Accumulate granulometry of 43 images has been computed.

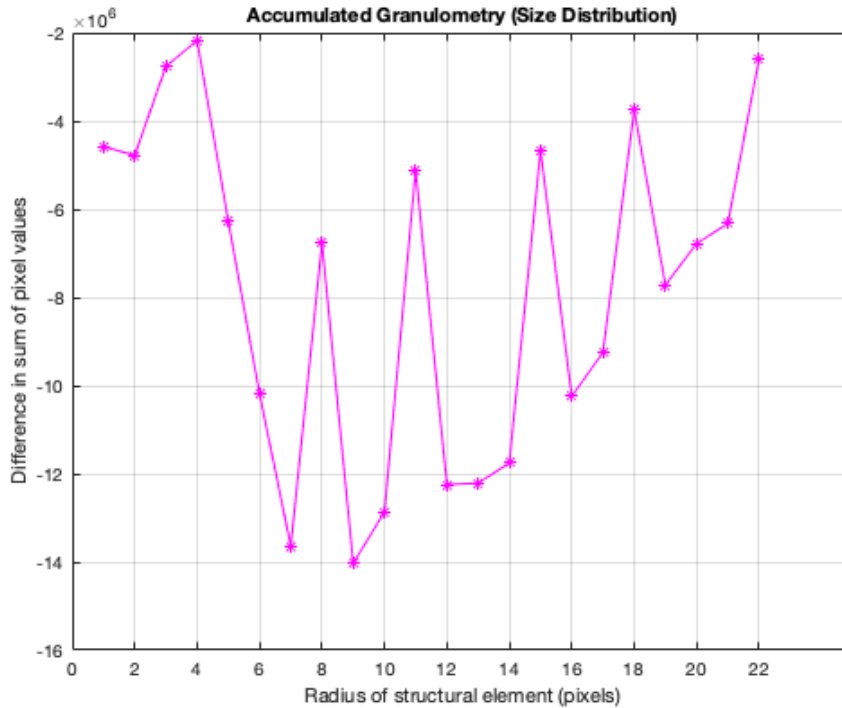


Figure 18: Granulometry of one image

It seems clear that something is happening when the opening by reconstruction of erosion is made with the original eroded image as marker using structuring elements which radius are 4, 8 and 11 pixels. In these values we are removing something that seems important. So, we are going to study these cases (not 11 because in 8 we see we start to remove relevant information).

- **Reconstruction with the original eroded image as marker using 4 pixels radius structuring element:** spurious maxima located in the mucus and variations in the grey level of the nuclei are removed, but at the same time we start to remove relevant data as some nuclei maxima.

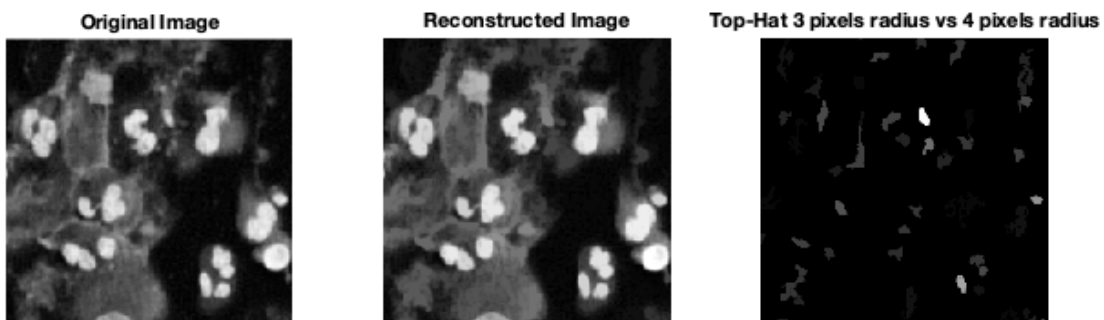


Figure 19: Closing vs Dual Reconstruction

- **Reconstruction with the original eroded image as marker using 8 pixels radius structuring element compared with 7 pixels radius structuring element:** relevant data as nuclei maxima are removed.

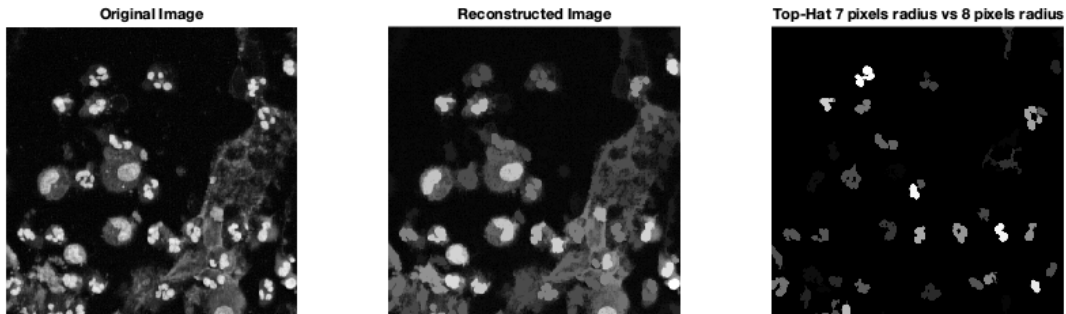


Figure 21: Top Hat 7 vs 8 pixels radius structuring element

Looking at the results, and taking into account that this software will be an intermediate step, which goal is to label images to train a CNN, calculation time is not the most important, we prioritize to keep relevant data. For that reason, we decided to clean the image applying reconstruction with the original eroded image as marker using 3 pixels radius structuring element. Which as we saw before, is very useful to reduce the number of nodes in the tree.

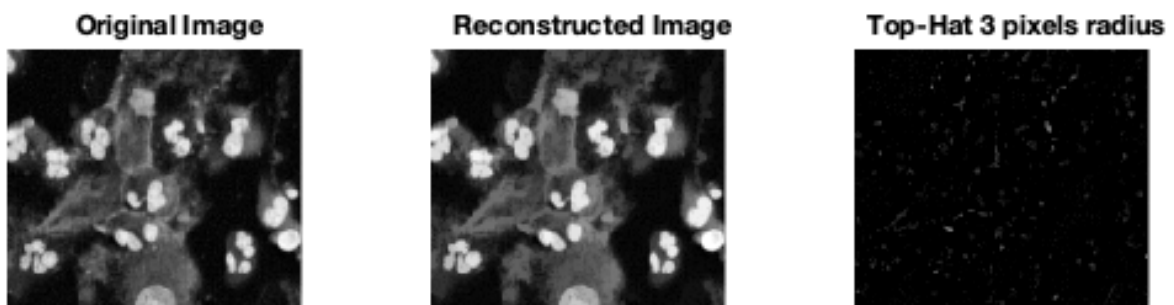


Figure 22: Reconstruction with 3 pixels radius structuring element

To clean minima, it is necessary to apply closing by dual reconstruction of image dilation, since with the closing, we obtain an unwanted overlapping. With the original image dilated as marker using 3 pixels radius disk structuring element, we manage to clean some noise in the background of the image without overlapping nuclei, so that was the selected technique.

3.2.2. Macrophage Detection

The first step is to compute the maxtree, which is an easy task thanks to maxtree processing toolbox [4].

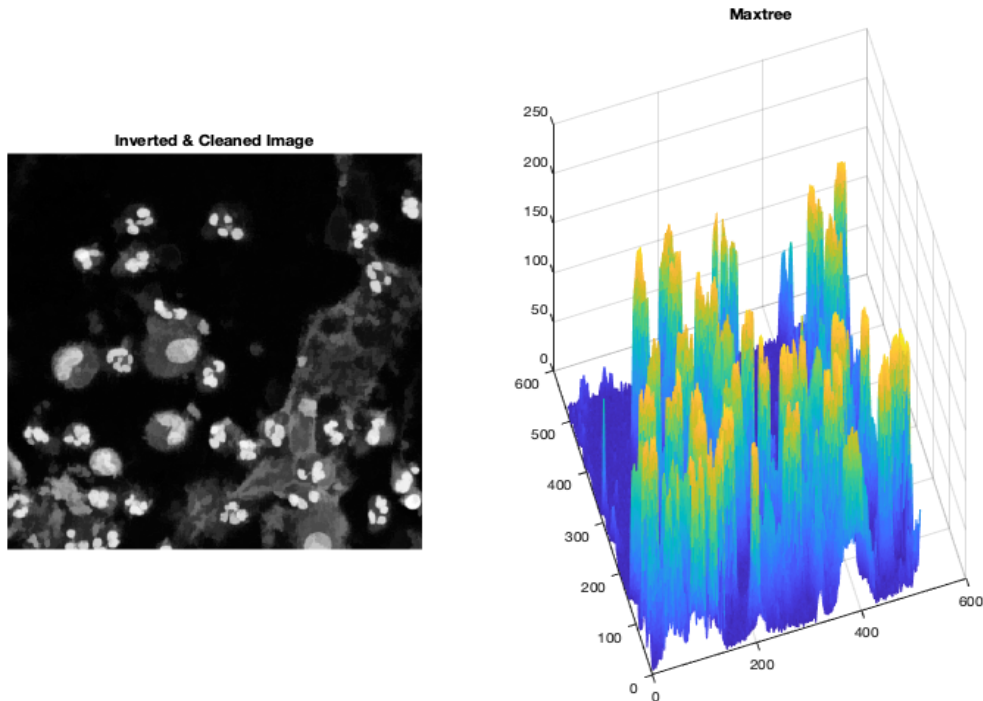


Figure 23: Maxtree Representation

The first analysed cells are macrophage, which we considered easiest for their shape and regularity. They always have the same structure, ellipse shape nucleus with cytoplasm. Taking this into account, two variable cuts in the trees have been done, one to determine the cytoplasm and another to determine the nuclei.

3.2.2.1. Macrophage cytoplasm, first cut

To decide whether or not a node belongs to macrophage cytoplasm 4 parameters are used, which are actually thresholds. If the node accomplishes them, the node is considered as macrophage cytoplasm node, if it does not accomplish them, it is not. The parameters are the following:

1. **min_leaf_glvl**: minimum leaf grey level that must have a node to be macrophage cytoplasm candidate.
2. **min_cito_glvl**: minimum grey level that must have a node to be macrophage cytoplasm candidate.
3. **max_cito_leafdist**: maximum grey level distance to the leaf that a node can have to be a macrophage cytoplasm candidate.
4. **min_cito_area**: minimum area that must have a node to be macrophage cytoplasm candidate.

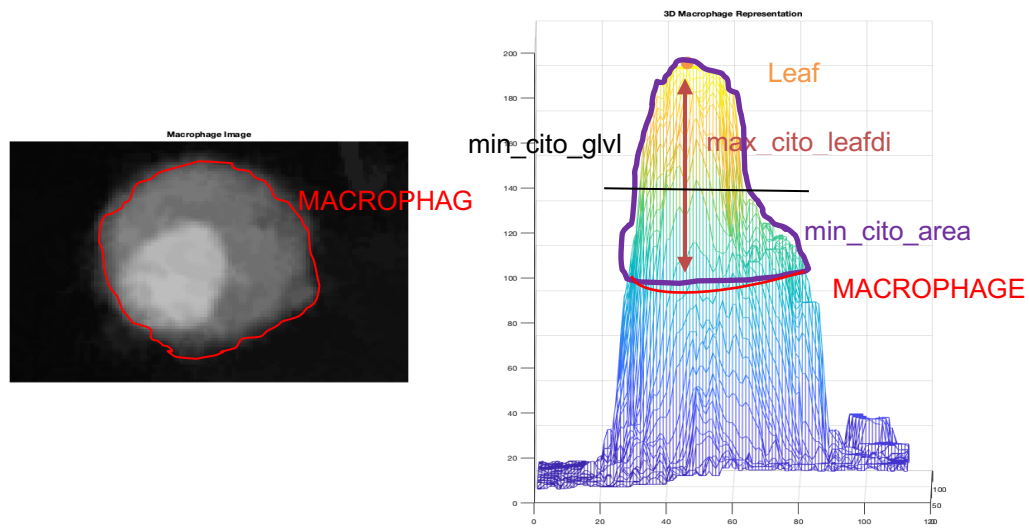


Figure 24: Macrophage cytoplasm analysis

This algorithm has the advantage of performing a local analysis of the nodes, since the assignment to a macrophage cytoplasm depends on the leaf, if a node does not present a grey level close enough to the leaf, it cannot be assigned as macrophage candidate. And if a node belongs to a branch which does not have a high enough grey level leaf, it cannot be assigned as macrophage candidate.

This makes that perhaps a macrophage with a cytoplasm which grey level is lower than some mucus, is detected as macrophage, because it may have a higher leaf and despite the fact that this grey level is lower than the common of macrophage cytoplasm, it can be assigned as macrophage cytoplasm candidate, since its classification depends on the distance to the leaf.

Another important feature is that cytoplasm candidate nodes must have a minimum area, understanding as area the sum of the number of pixels stored in the current node and all its descendant nodes [4]. This is because the rest of cells are smaller than macrophages, and the rest of parameters do not filter them, so some of the other cell types would be determined as macrophages. For that reason, we need to filter these candidates with little area. We do not use a parameter to filter how big a macrophage cytoplasm, this is because in some images appears what is called “soups of macrophage”, where many macrophages appear sharing a common cytoplasm. This cytoplasm is very big, and these macrophages have to be counted as well.

Finally, the last parameter is to filter the nodes which have a very low grey level, since some mucus can be detected as macrophage if it has spurious maxima and enough area.

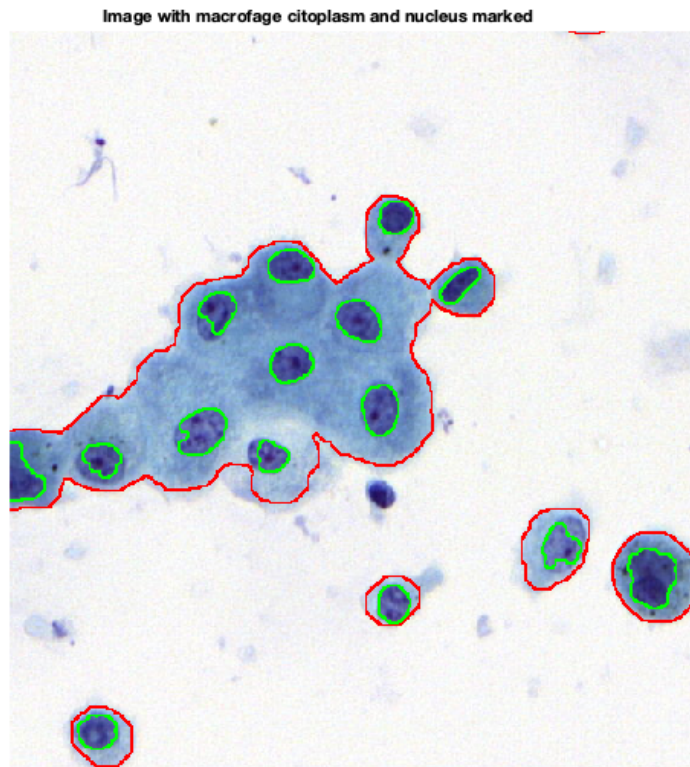


Figure 25: Macrophage Soup

All the parameter values have been selected by hand, testing many images and choosing those ones that best determine the correct candidates.

- **Final values chosen**

min_leaf_glvl	min_cito_glvl	max_cito_leafdist	min_cito_area
120	30	0.25	1400

Table 4: Parameters final values

3.2.2.2. Macrophage nuclei

Once a macrophage cytoplasm is detected, we tried to determine which nodes are macrophage nuclei. For this task, the same technique used with macrophage cytoplasm candidates will be used, changing some parameters and their values.

1. **min_leaf_glvl**: minimum leaf grey level that must have a node to be macrophage nucleus candidate.
2. **min_nuc_glvl**: minimum grey level that must have a node to be macrophage nucleus candidate.
3. **max_nuc_leafdist**: maximum grey level distance to the leaf that a node can have to be a macrophage nucleus candidate.

4. **min_nuc_area**: minimum area that must have a node to be macrophage nucleus candidate.
5. **max_nuc_area**: maximum area that must have a node to be macrophage nucleus candidate

- **Final values chosen**

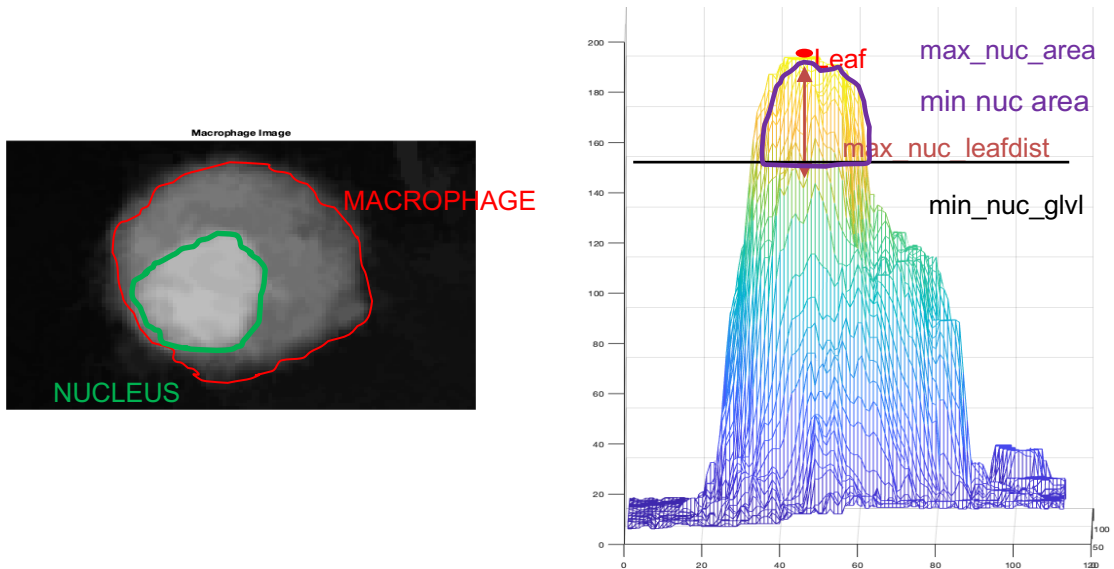


Figure 26: Macrophage nucleus analysis

min_leaf_glvl	min_nuc_glvl	max_nuc_leafdist	min_nuc_area	max_nuc_area
160	50	0.74	200	2000

Table 5: Macrophage nuclei parameters final values

- **Detection Example**

In the next figure we can see an example where two macrophages are detected and contamination is rejected, because it does not have enough area in the cytoplasm cut.

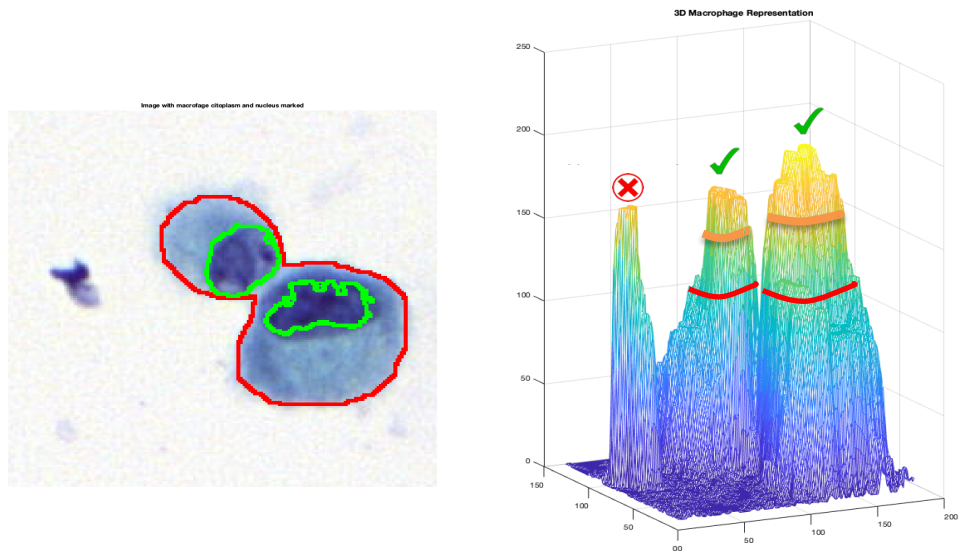


Figure 27: Macrophage Detection Example

3.2.2.3. From candidate nodes to candidate pixels

Before this classification algorithm, we have the nodes that the algorithm considers they belong to macrophage or macrophage nuclei. A recovery of the pixels associated to these nodes is necessary. For this task, a function of the maxtree processing toolbox [4] is used, which returns an image where pixels associated to nodes that are marked as candidates have their branchID (branch identifier, each branch has one) and those pixels associated to nodes which are not marked as candidates have 0 as value.

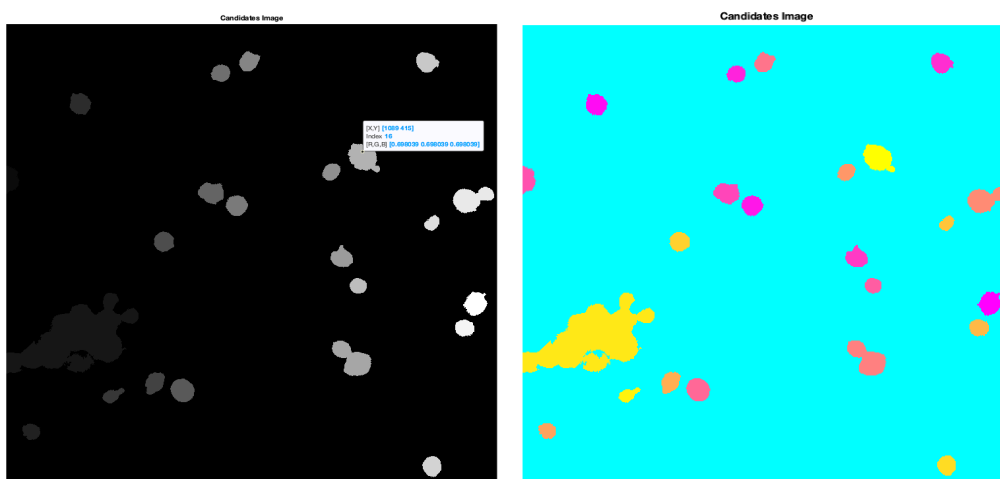


Figure 28: Macrophage cytoplasm image of candidates

3.2.2.4. Images of candidates cleaning

Sometimes, images of candidates are not as good as we want; they are noisy, with some overlapped cells, with holes inside candidates... In order to improve the quality of these reconstructed images, post cleaning techniques are applied.

- **Macrophage cytoplasm image of candidate closing by dual reconstruction of image dilation**

To implement hole filling without removing important data, as minimums which separate two or more close macrophages, we use dual reconstruction with the original image of candidates eroded using 10 pixels radius disk structuring element. We have chosen it instead of opening because it better preserves the shapes and does not print the structuring element on the image.

- **Macrophage cytoplasm image of candidates opening**

To clean some spurious macrophages that have a lot of area because they are long and narrow, we applied opening with 15 pixels radius disk as structuring element. In addition, with this opening we separate some different cells that are a bit overlapped (it is very common to find a macrophage and a lymphocyte overlapped, with this opening, we solve this problem). In this case, a reconstruction step should not be used, because by reconstruction overlapped cells are not separated.

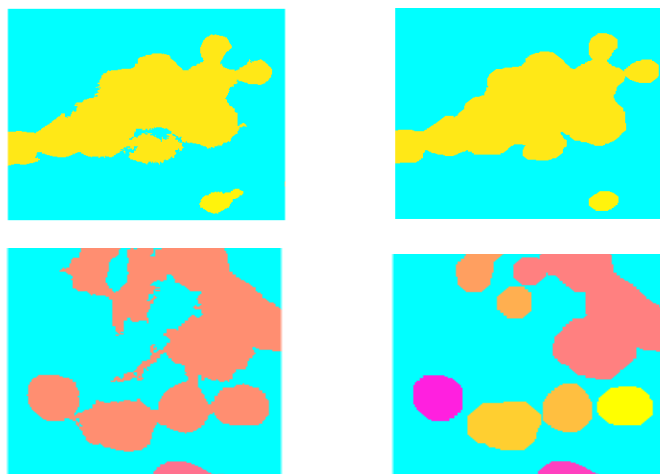


Figure 29: Macrophage cytoplasm opening and dual reconstruction

- **Macrophage nuclei image of candidates closing by dual reconstruction of image dilation**

Like before, we want to do hole filling without removing important data, such as minima which separate two or more close nucleus. For that reason, dual reconstruction with the original image of candidates eroded using 10 pixels radius disk structuring element has been applied.

- **Macrophage nuclei image of candidates opening**

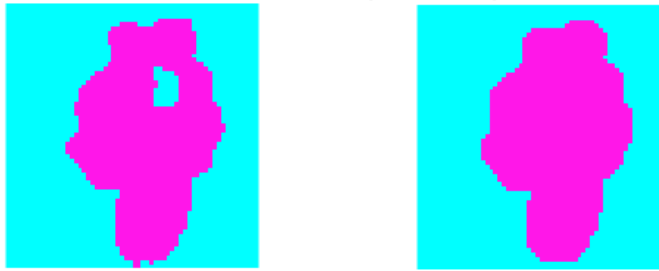


Figure 30: Macrophage nucleus hole filled

In order to clean some spurious macrophages nucleus and separate some others that are a bit overlapped, opening with 8 pixels radius disk as structuring element has been applied. Finally, macrophage nucleus detected outside macrophage cytoplasm are automatically rejected.



Figure 31: Macrophage nucleus cleaned

- **Image of candidates cleaning examples**

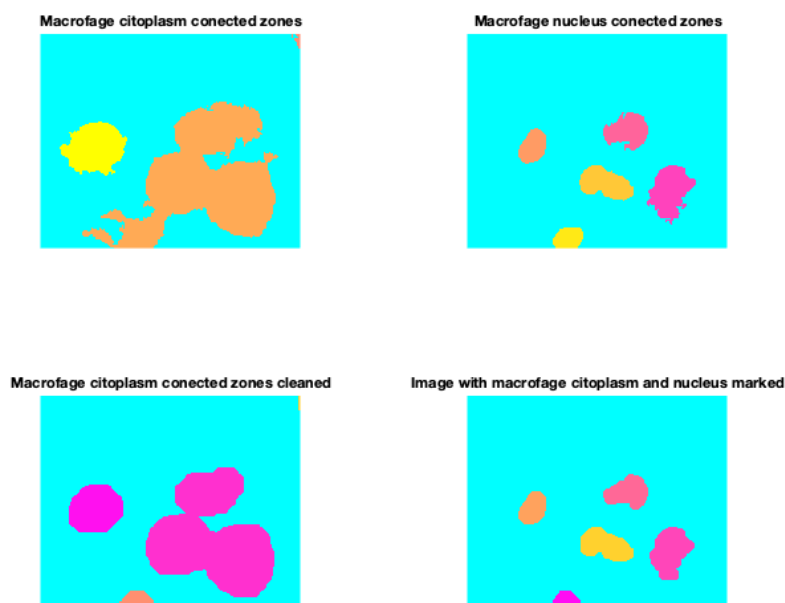


Figure 32: Macrophage image of candidates cleaning example 1

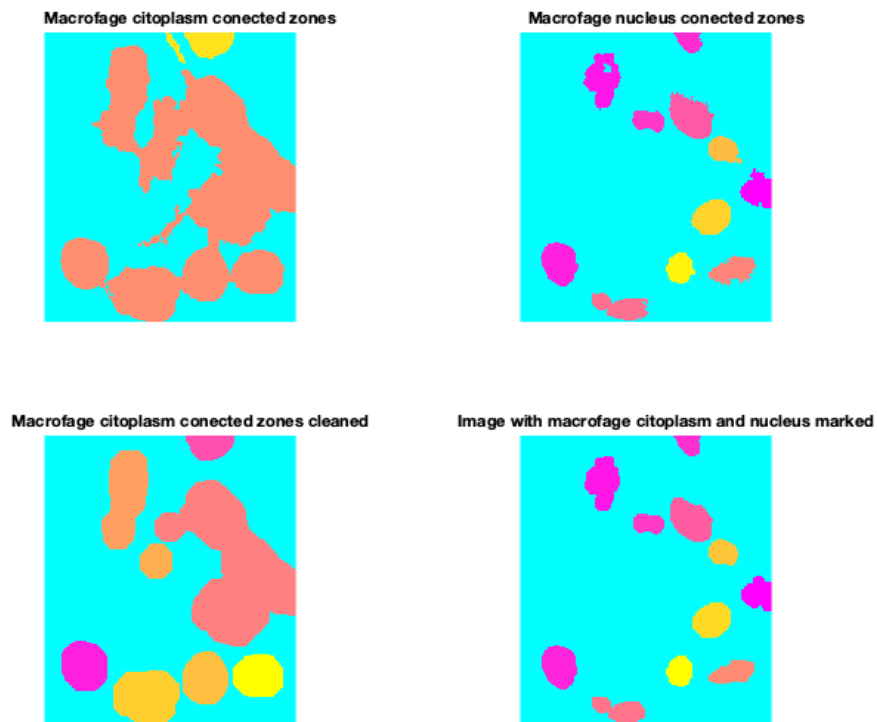


Figure 33: Macrophage image of candidates cleaning example 2

3.2.2.5. Macrophage detection

To decide which pixels mark as macrophages or which ones not, we had two options: mark as macrophage pixel the cytoplasm candidates, or mark as macrophage pixel the nuclei candidates. As the application final goal is to count the number of every cell type present in the image to give a medical diagnosis, we decided to mark as macrophage only the pixels that belong to nuclei candidates. Because it is likely that two or more macrophage overlap, so they share cytoplasm and the only way to differentiate them is through their nuclei. We can see a clear example in Figure 25.

- **Macrophage detection examples**

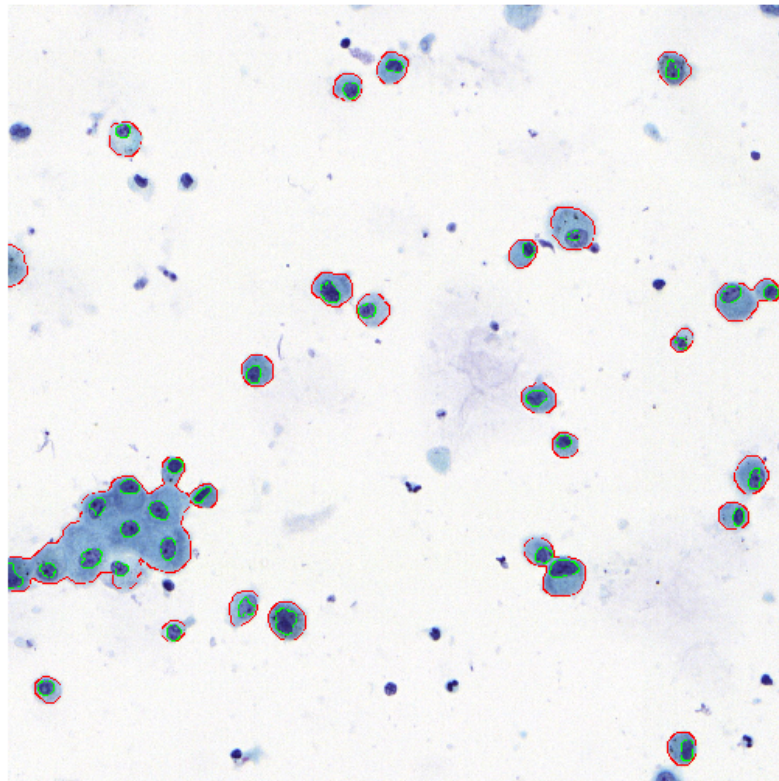


Figure 34: Macrophage detection example 1

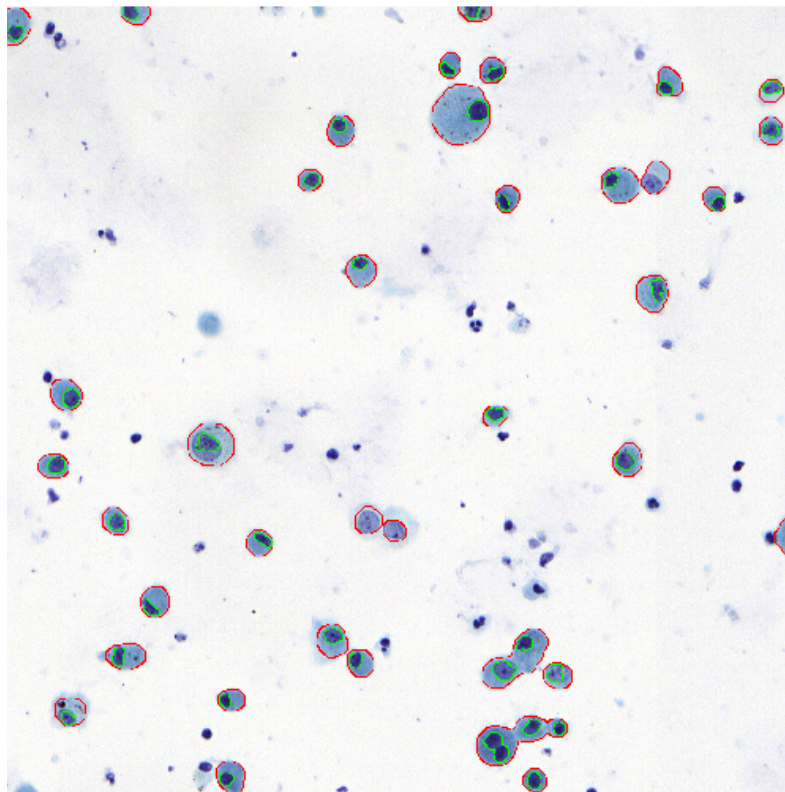


Figure 35: Macrophage detection example 2

3.2.3. Lymphocytes, Polymorphonuclear and Eosinophiles Detection

These three types of cells have been grouped because their size is very similar, it depends more on the chosen BAL sample than on the type of cell. For that reason, we decided firstly, detect these three types of cells together, making a cut in the tree, and once they are detected, separate them making another cut.



Figure 36: Polymorphonuclear / Eosinophile / Lymphocyte

3.2.3.1. First maxtree cut, looking for cytoplasm, joint detection

As we do in macrophage detection, we are going to take advantage of the Maxtree analysis to select lymphocytes, polymorphonuclears and eosinophiles candidates. So, the first step is to compute the Maxtree.

Once this step is done, the same algorithm used in the macrophage cytoplasm is applied, we do a variable cut in the tree, following a series of parameters, which will determine if a node is considered as candidate or not.

1. **min_leaf_glvl**: minimum leaf grey level that must have a node to be considered as candidate.
2. **min_cyt_glvl**: minimum grey level that must have a node to be considered as candidate.
3. **max_cyt_leafdist**: maximum grey level distance to the leaf that a node can have to be considered as candidate.
4. **min_cyt_area**: minimum area that must have a node to be considered as candidate.
5. **max_cyt_area**: maximum area that must have a node to be considered as candidate.

It is exactly equal that macrophage cytoplasm detection, changing parameters. Figure 24

- **Final values chosen**

min_leaf_glvl	min_cyt_glvl	max_cyt_leafdist	min_cyt_area	max_cyt_area
145	50	0.25	50	600

Table 6: Small cells cytoplasm parameters final values

3.2.3.2. Second maxtree cut, looking for cytoplasm, joint detection

Now, following as we did in the macrophages, we intend to make a second variable cut in the tree, trying to find the nuclei candidates. The used values for the parameters are the following:

1. **min_leaf_glvl**: minimum leaf grey level that must have a node to be considered as candidate.
2. **min_nuc_glvl**: minimum grey level that must have a node to be considered as candidate.
3. **max_nuc_leafdist**: maximum grey level distance to the leaf that a node can have to be considered as candidate.
4. **min_nuc_area**: minimum area that must have a node to be considered as candidate.
5. **max_nuc_area**: maximum area that must have a node to be considered as candidate.

It is exactly equal that macrophage cytoplasm detection, changing parameters. Figure 26

- **Final values chosen**

min_leaf_glvl	min_nuc_glvl	max_nuc_leafdist	min_nuc_area	max_nuc_area
185	170	0.89	20	300

Table 7: Small cells nuclei parameters final value

- **Detection Example**

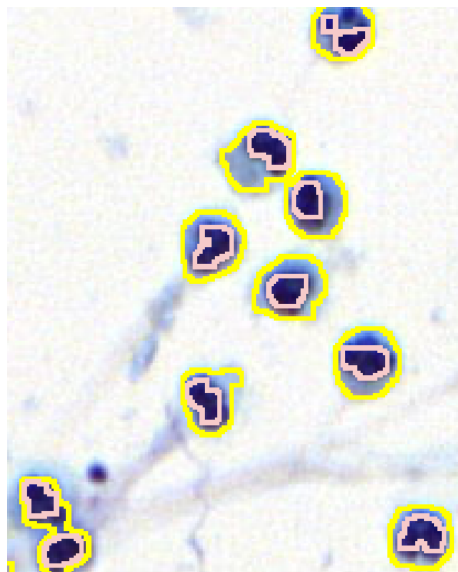


Figure 37: Lymphocytes, polymorphonuclears and eosinophiles detection

This cut, besides providing us the pixels which belong to nuclei, should provide us the necessary information to differentiate cells; considering that, as we have seen, lymphocytes only have one nucleus, eosinophiles two, and polymorphonuclear three or more. But actually, we saw that in the vast majority of cases, nuclei are overlapped, and cleaning the image will be not enough to differentiate the different type of cells.

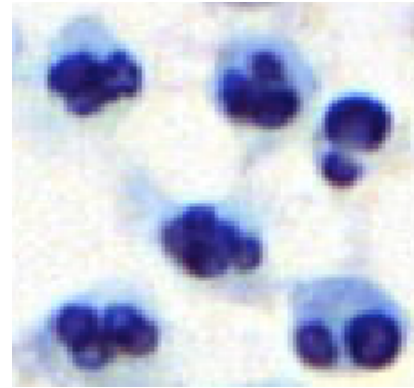
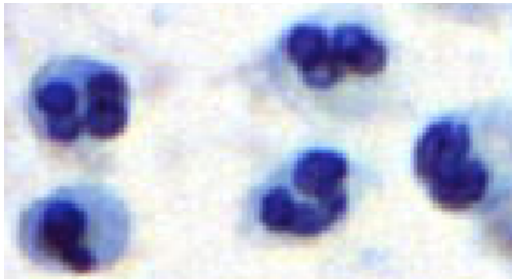


Figure 38: Nuclei Overlapped

3.2.3.3. Images of candidates cleaning

To generate candidate images, we use the same technique used in macrophage detection, and equal to macrophages case, images of candidates are not as good as we want. We only did a small opening by reconstruction with the original image of candidates dilated as marker using a 3 pixels radius disk structuring element to remove spurious maxima in mucus. Nuclei detected outside cytoplasm, are automatically rejected. We also reject the candidates detected inside macrophage nuclei, because we consider that macrophage nuclei are better detected, and it is very rare to find a lymphocyte, polymorphonuclear or eosinophile above a macrophage.

The real goal in these cells cleaning is to achieve separate nuclei, because this will be the way to differentiate the cell type.

We have tried with openings, using different sizes of structuring elements. The size we saw we obtained acceptable results and it did not remove relevant data is 3 pixels radius structuring element, because with 4 pixels we start to lose relevant data.

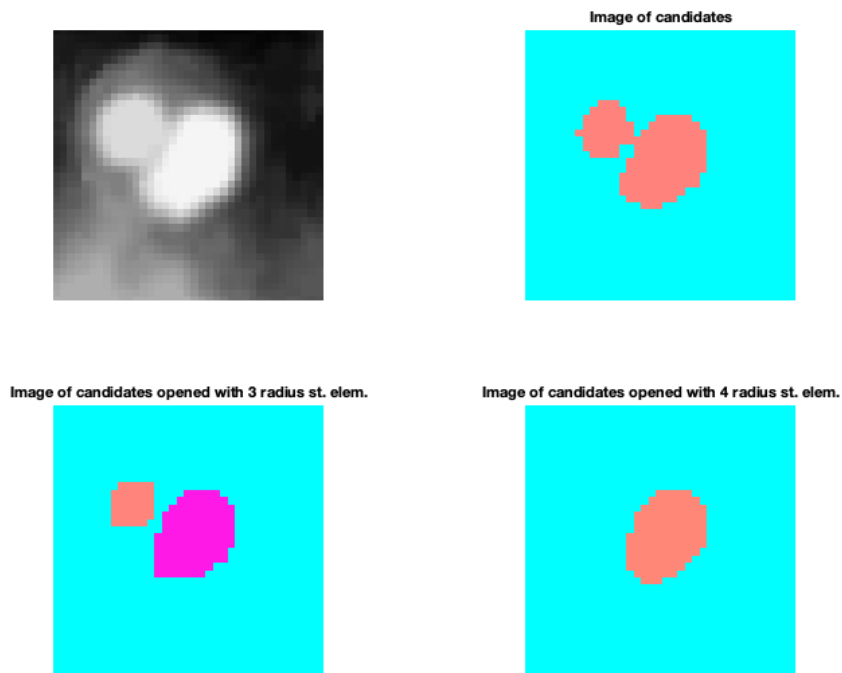


Figure 39: Example of how 4 pixels radius structuring element remove relevant data

And here we have a successful example, but they are isolated cases. The most common is not to achieve good separation.

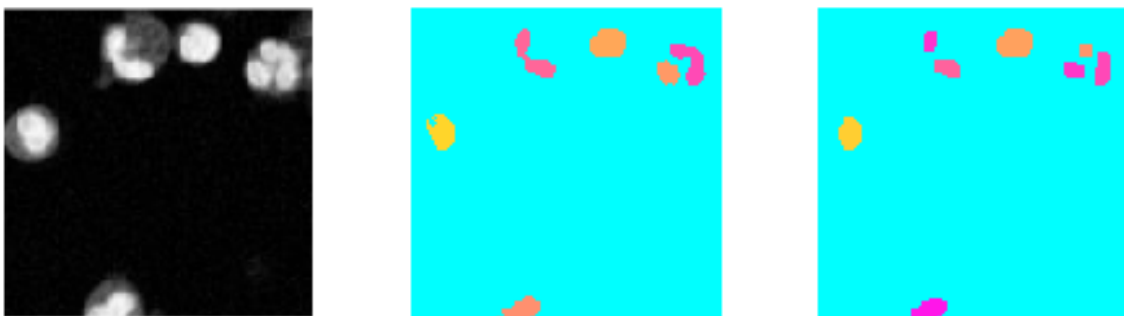


Figure 40: Second cut example

This was a serious problem, because we cannot differentiate one type of cell from the others, we needed to find a way to do it. At first, we think about doing a third cut into the tree, but we have tried and the cut was not very accurate, because leaves were not in every nucleus, and not always a maximum was related to a nucleus.

3.2.3.4. Distance Function

The distance transform provides a metric or measure of the separation of points in the image. It calculates the distance between each pixel that is set to off (0) and the nearest nonzero pixel for binary images. In our case, we use Euclidean distance [10].

It can be computed with cascading erosions, as we can see in Figure 41.

Taking into account that the maxtree solution does not seem to help us in this task another option is to find a solution based in the shape of nuclei, implementing a distance function solution. Our intention was to find the maxima in the distance function of the image of nuclei candidates, and this would return the number of nuclei in a cell.

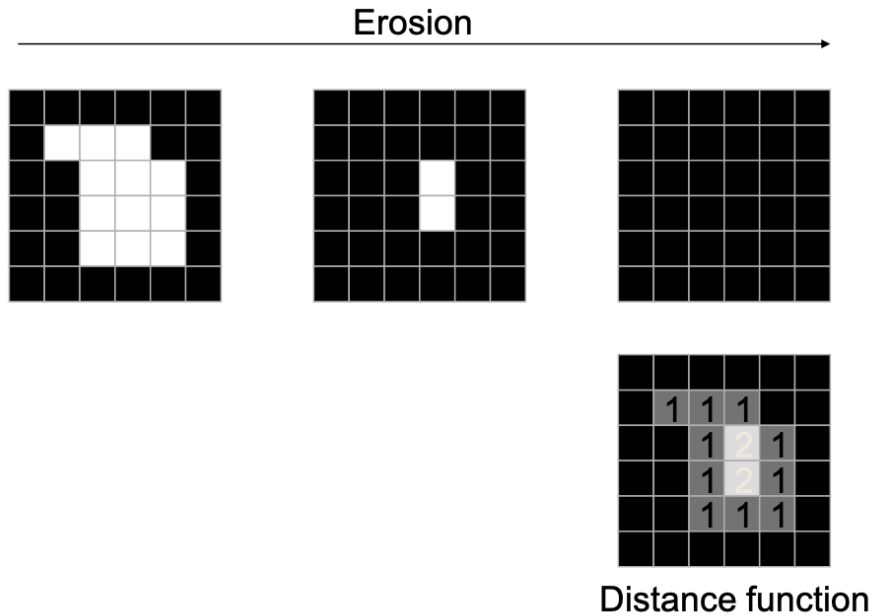


Figure 41: Distance Function

Here we have some results, where Maxima1 are maxima of the distance function calculated with a very restrictive threshold, whereas Maxima2 are maxima of the distance function calculated with a less restrictive threshold. These maxima are computed using maxtree cuts, such as in cell detection, but in this case, there are almost no nodes.

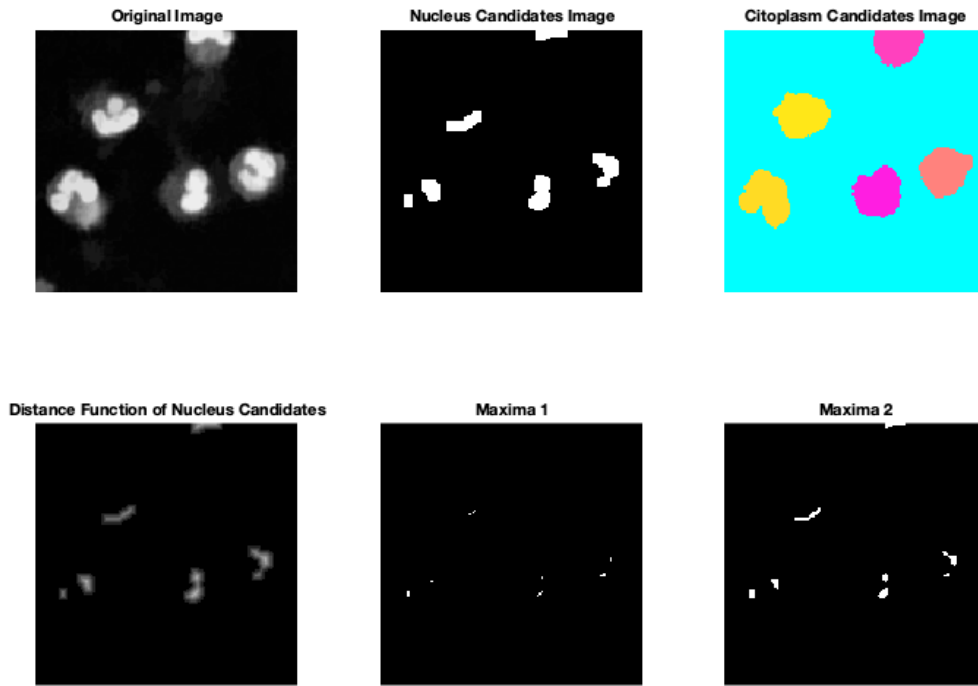


Figure 42: Analysis by maxima of distance function 1

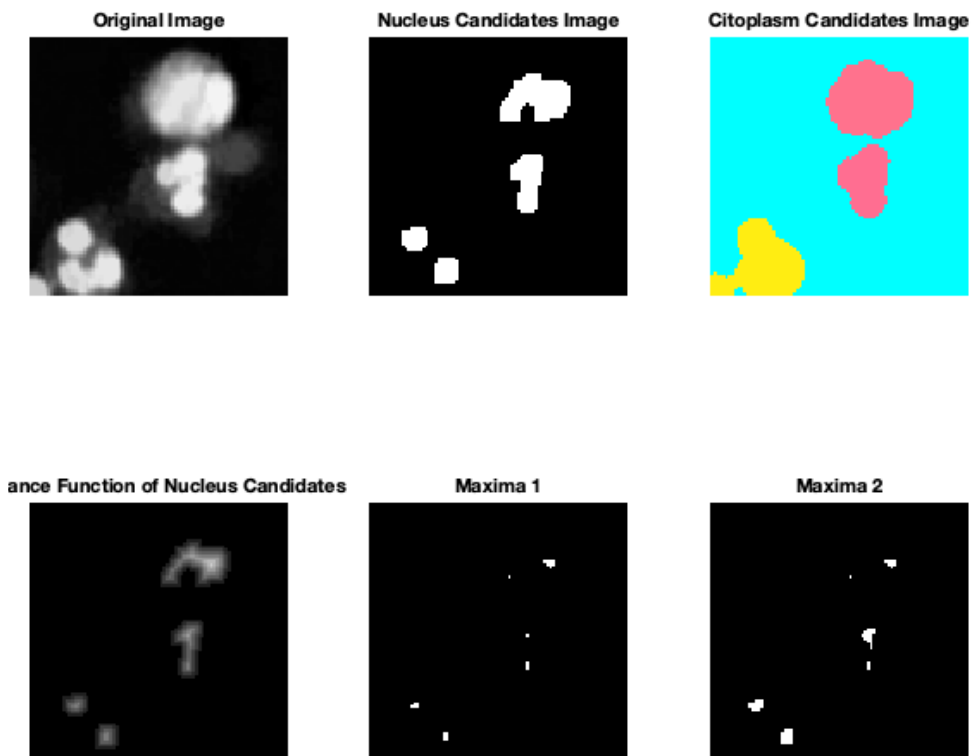


Figure 43: Analysis by maxima of distance function 2

The proposed technique is not always effective. It improves the separation of the cleaning techniques but it continues confusing the cell type. This was the final way we used in the classical algorithm to differentiate cells counting the number of connected components coming out of the maxima of distance function inside the cells. If it has one connected component is a lymphocyte, if it has two it is eosinophile and if it has more it is polymorphonuclear. If the cell candidates do not have any maxima of distance function, they are rejected. I want to remark that in this case, the cells are marked with cytoplasm, because they can have more than one nucleus.

The results in small cells (lymphocytes, polymorphonuclears and eosinophiles) are not as good as in the macrophage case, because as we said before, the algorithm continues confusing the cell type. It seems that looking for maxima in the image is not the best way to separate the overlapped nuclei, because not all maxima in the image are nuclei and not all nuclei have the maximum of the cell inside. As we have seen, it was a difficult task that we underestimated. However, the cell cytoplasm is quite well segmented, the problem is how to differentiate one type of cell from the others. More research can be done to achieve this classification using classical techniques (as proposed in future development or others). Nevertheless, the results obtained could already be used as a preliminary ground truth for training a machine learning algorithm.

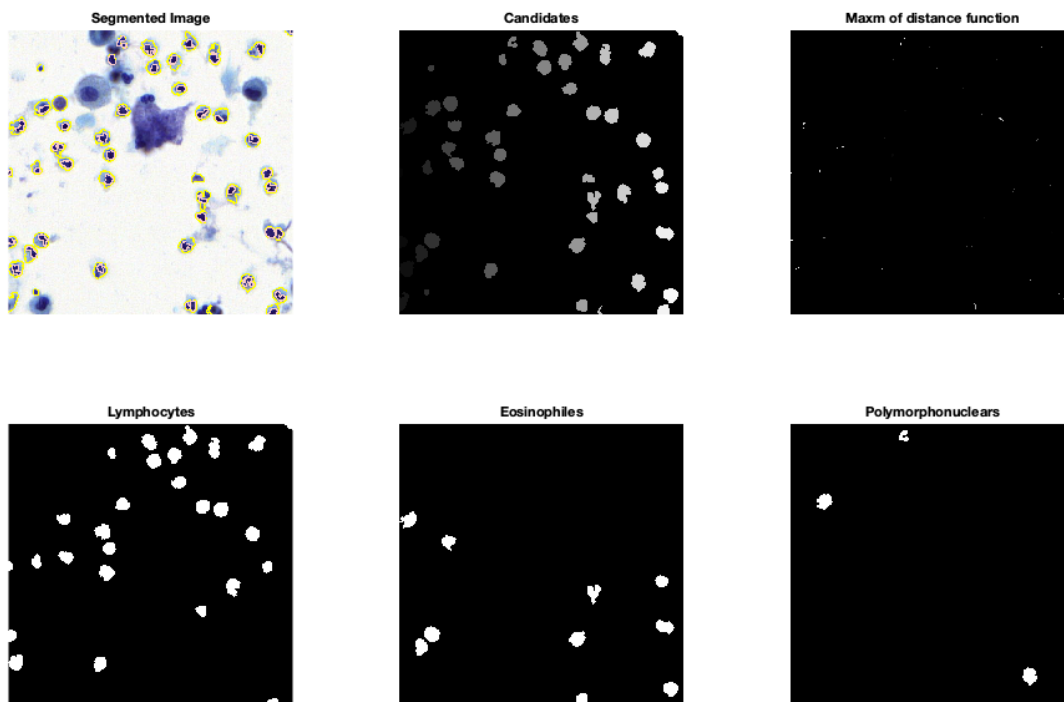


Figure 44: Example of small cells detection 1

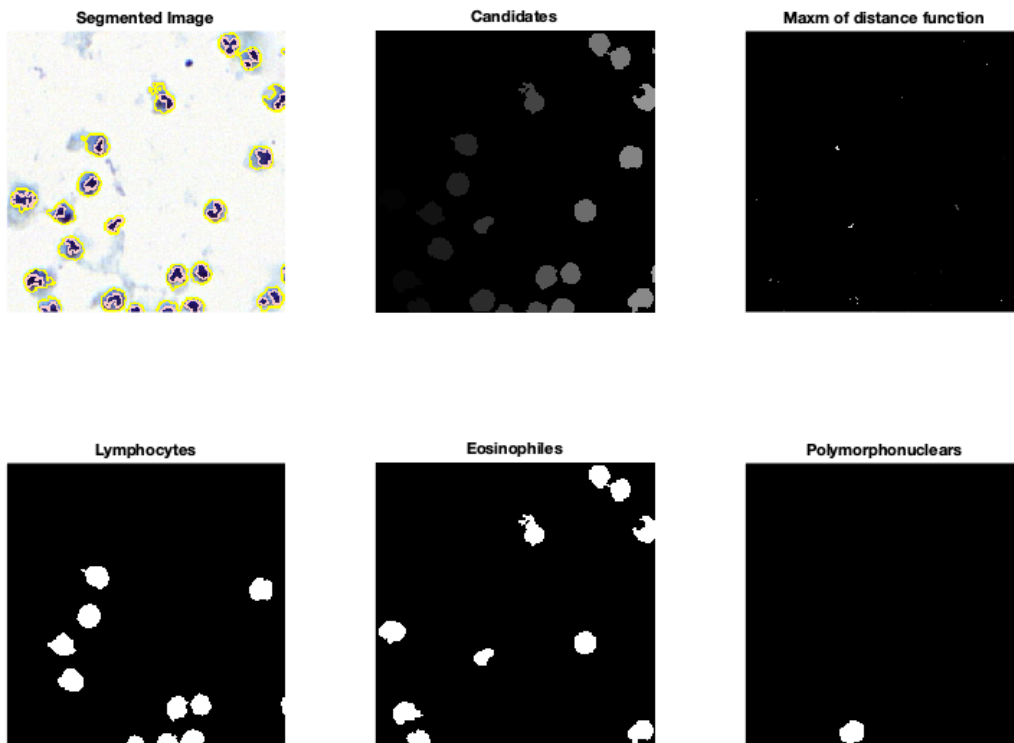


Figure 45: Example of small cells detection 2

3.2.4. Mask creation

Once we have applied all these techniques, masks have to be generated. The algorithm generates as output grey scale images, where 0 value pixels are background, 64 value pixels are macrophage, 128 value pixels are lymphocytes, 191 value pixels are polymorphonuclears and 255 value pixels are eosinophiles.

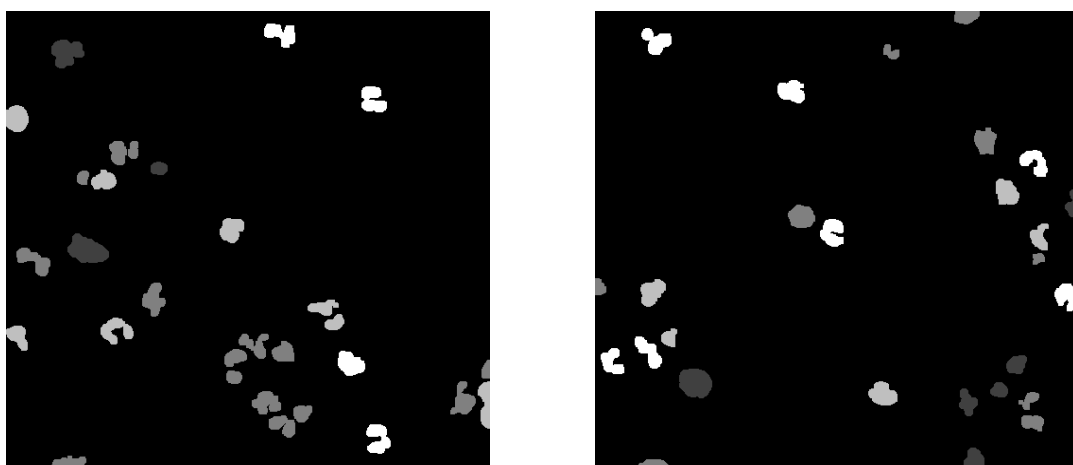


Figure 46: Output masks examples

3.2.5. Manual Labelling Corrections

To solve the classification problem of the small cells and to improve the results of the maxtree, we decide to correct by hand the output masks, in order to train a small CNN to start testing how they work with this problematic. If it works better than the maxtree algorithm, we could continue labelling images with the output of the CNN and correcting them by hand, but it is always easier to correct something more or less good than correct something from zero.

3.2.6. VGG Image Annotator

Once an algorithm segments the images in the different classes and background, the segmentation has to be corrected. For that reason, a tool to correct the labelled images is needed. We use the VGG Image Annotator (VIA), a simple and standalone manual annotation software for image, audio and video. VIA runs in a web browser and does not require any installation or setup. The complete VIA software fits in a single self-contained HTML page of size less than 400 Kilobyte that runs as an offline application in most modern web browsers [11].

With this annotator, we can generate easily new annotations with so many different shapes, but if you have previous annotations, the only way to load them in the browser is convert all the regions in a recognizable shape (square, rectangle, circle, etc.), in our case, ellipses are the defined shapes that best suit our cells. After that, we should write them in a .json file, to be able to modify the masks. And loading this .json file in the browser annotator we can start to correct our labelled images.

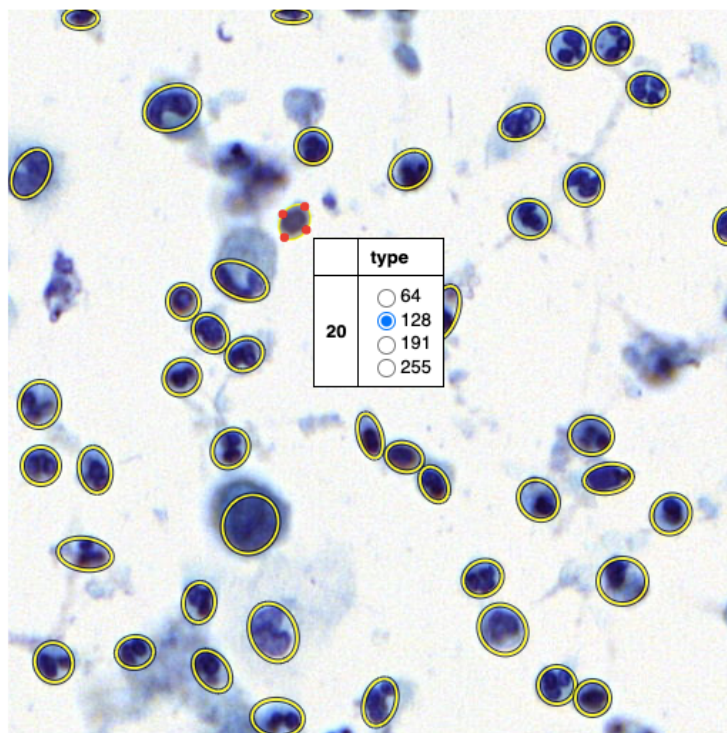


Figure 47: Correcting masks in the VGG Annotator

Once we have the masks corrected, we want to get the shapes back, but this is not possible directly, because this annotator returns the ellipses. In order to get back the regions, we have to apply a watershed step.

All the algorithms needed to make the correction were provided to me by my tutors and external people, because they were applied in another tasks. The part we have to do was to understand how they work and modify some parameters and processes to adapt the algorithms to our problematic.

3.2.6.1. Corrections

It is important to say that these corrections have been done with some supervision of the doctor Jordi Temprana, who helps us to segment correctly some doubtful images and cells.

We generate a dataset of 56 images with their corresponding masks, at first segmented with our algorithm and finally corrected with the VGG Annotator.

As you already know, we have a classification problem with the small cells, so the most frequent error corrected is the change of class, where the detection is well, but the classification does not.

The procedure to do that is the following:

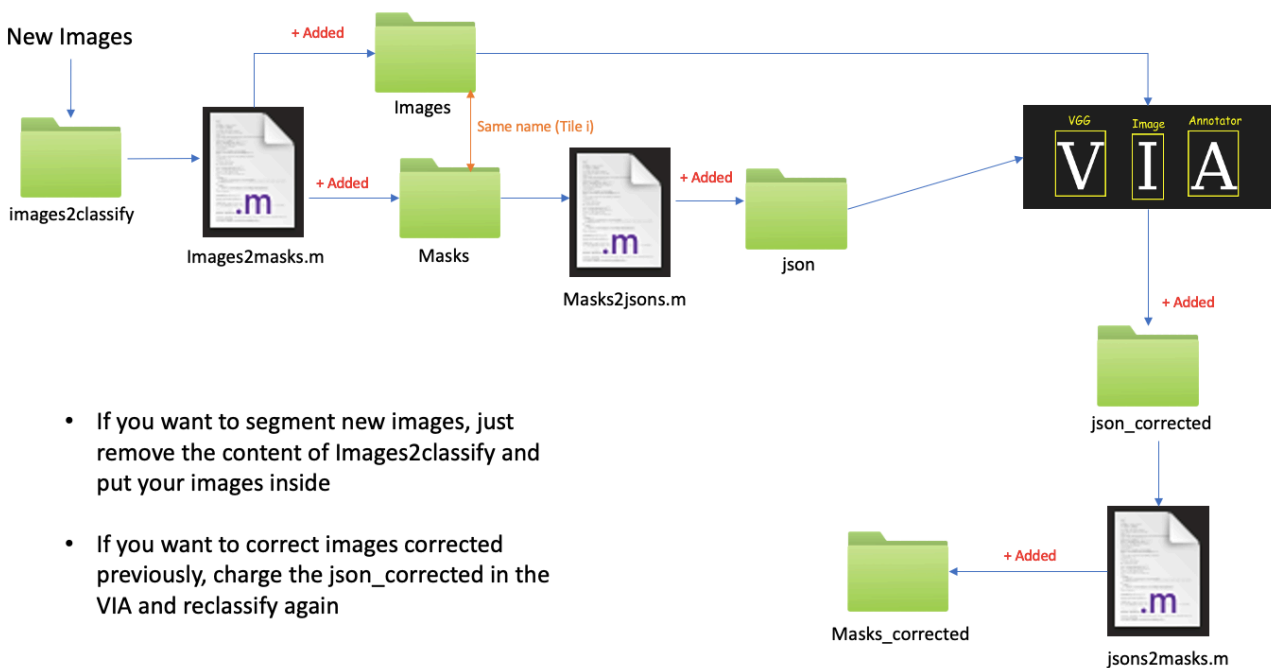


Figure 48: Images Segmentation Procedure

- **Correction Examples**

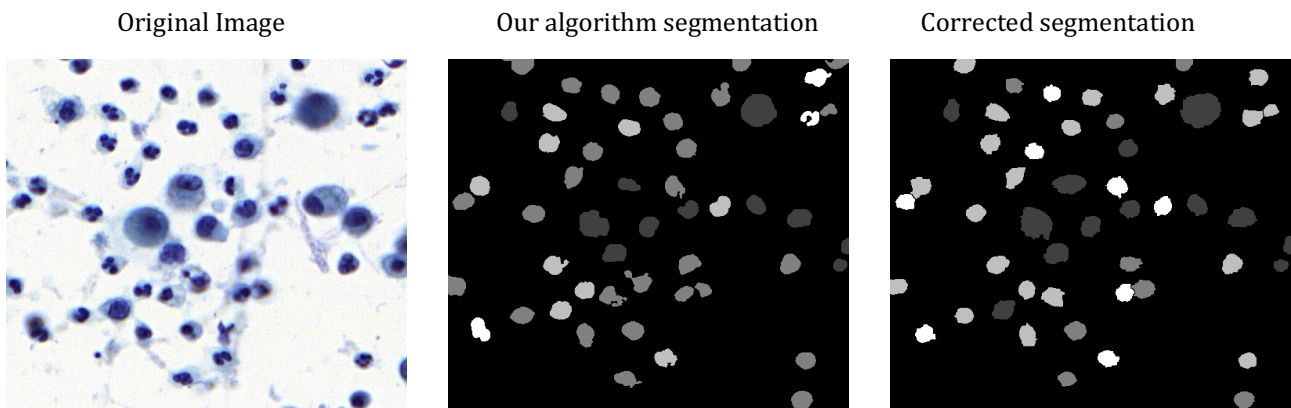


Figure 50: Correction example 1

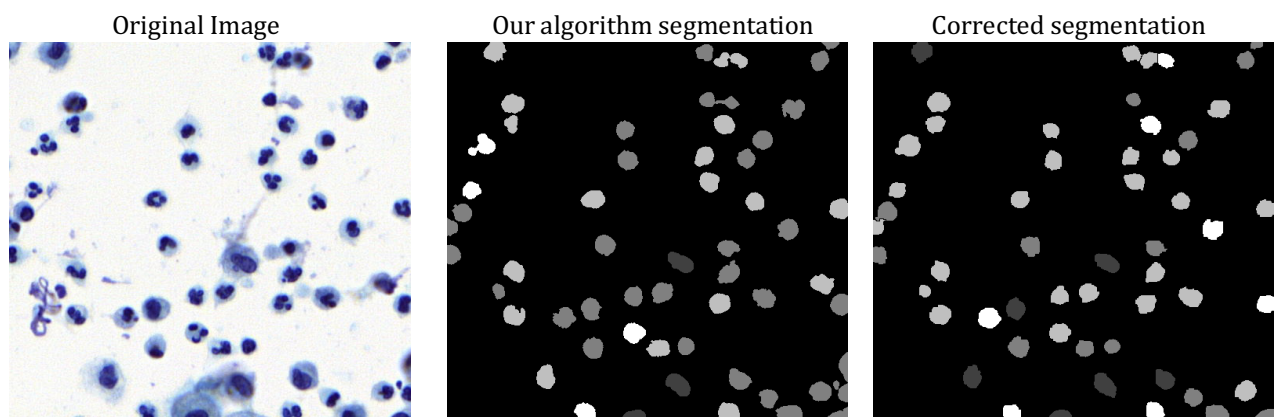


Figure 49: Correction example 2

3.3. Convolutional Neural Network, U-Net

3.3.1. Introduction

There is large consent that successful training of deep networks requires many thousand annotated training samples [9]. In our case we only have 56 annotated images, which will result in 11 validation samples (20%) and 45 training samples (80%).

In number of cells, we have annotated 1.526 cells, which breakdown is: 388 macrophages, 312 lymphocytes, 782 polymorphonuclears and 44 eosinophiles.

The goal of this CNN is not to generate the final software for the image segmentation, which gives the definitive results. We intend to make a test with a small dataset, a prove of concept to see how CNN manage with these images, and if it is acceptable, take ideas or use it to generate more images for the dataset.

Network and training strategies that relies on the strong use of data augmentation have to be applied to use the available annotated samples more efficiently. It is generated applying elastic deformations to the available training images. This allows the network to learn invariance to such deformations and avoid overfitting with small datasets, without the need to see these transformations in the annotated image corpus. This is particularly important in biomedical segmentation, since deformation used to be the most common variation in tissue and realistic deformations can be simulated efficiently [9].

To carry out this CNN segmentation (https://github.com/qubvel/segmentation_models.pytorch) [12] library has been used.

3.3.2. Network Architecture

We have used the U-Net architecture to carry out the CNN development.

3.3.3. Data Augmentation

How do I get more data, if I don't have "more data"? Knowing that CNN are not smart, to get more data, we just need to make minor alterations to our existing dataset. Minor changes such as flips or translations or rotations. Our neural network would think these are distinct images anyway [13]. To do this task, the library (<https://github.com/albumentations-team/albumentations>) [14] have been used.



Figure 51: Data Augmentation in play [13]

3.3.4. Transfer Learning

As we have little labelled data, we use transfer learning as [15] [12] suggest. Essentially, the net starts the learning process from patterns that have been learned for a different but similar task, instead of starting the learning process from a (often randomly initialized) blank sheet.

Also, humans are not taught every single task or problem in order to be successful at it. Everyone gets into situations that have never been encountered, and we still manage to solve problems in an ad-hoc manner. The ability of learning from a large number of experiences, and exporting 'knowledge' into new environments is exactly what transfer learning is all about [16].

Following [15] [12], ResNext encoder, specifically *resnext50_32x4d* have been used as backbone, with Imagenet pre-trained weights.

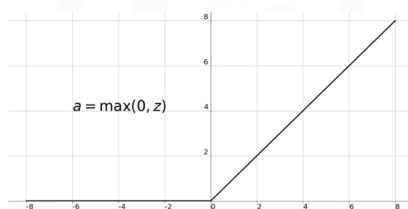
3.3.5. Training mini-batch size

As we have 45 images for training, 5 images mini-batch is used, in order to use the same number of images in every epoch and do not give more relevance to some images, because if the last mini-batch is smaller, its images would have more influence in weights if we do not discard them (drop last).

3.3.6. Activation Function

The activation functions are the way to introduce non-linearities in the network, it defines the output of a node given a set of inputs. For that purpose, two different activation functions have been tested in the CNN, ReLU and sigmoid.

ReLU Function



Sigmoid Function

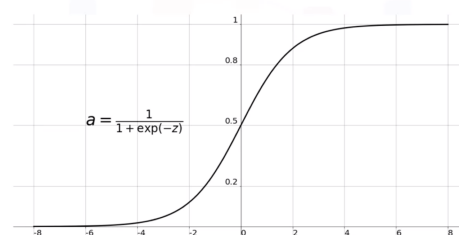


Figure 52: ReLU vs Sigmoid [28]

Sigmoid was the activation function finally chosen, because independently of the rest of the parametrization it presents better results than ReLU. IoU scores with sigmoid are in average 5 – 10 % greater than with ReLU, independently of the number of epochs, learning rate or optimizer. A possible reason may be that sigmoid suits better our data to converge towards a better solution.

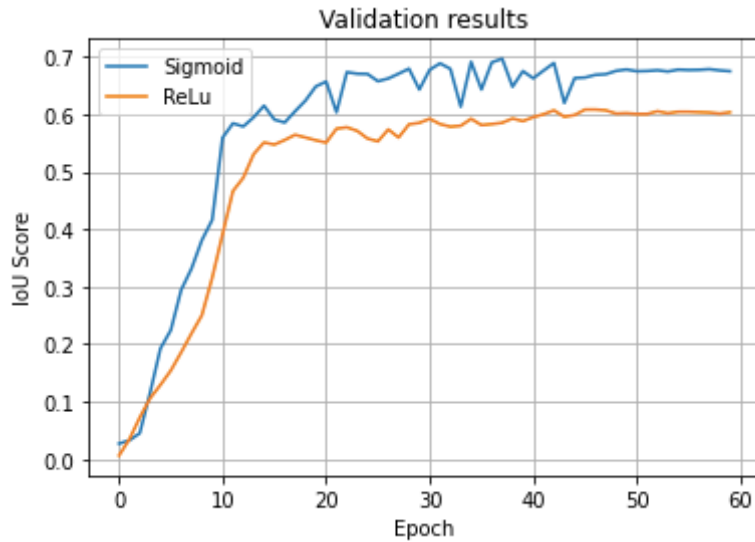


Figure 53: Sigmoid vs ReLU

3.3.7. Classification

As output of the network, we obtain 4 feature maps, one for each class. In feature map i there are the probabilities of each pixel to belong to the class i . These probabilities are computed by a pixel-wise soft-max:

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

$p_k(y = k | \mathbf{x})$ is the probability that pixel y belongs to class k .

$a_k(\mathbf{x})$ is the activation in feature channel k at the pixel y

So, if $p_k(y = k | \mathbf{x}) \approx 1$ for the k that has the maximum activation $a_k(\mathbf{x})$ and $p_k(y = k | \mathbf{x}) \approx 0$ for all other k .

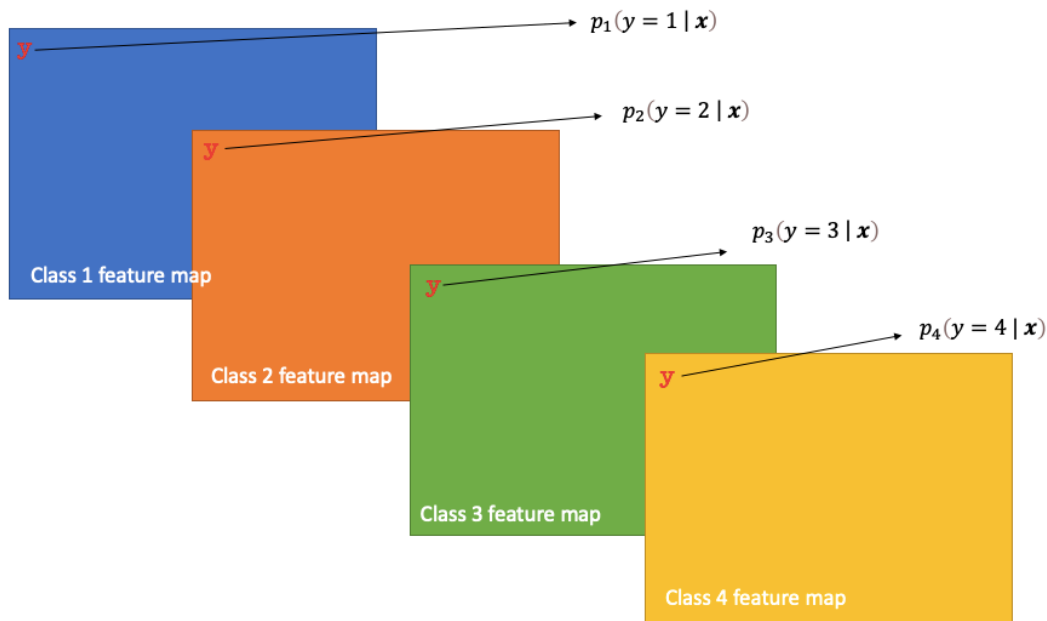


Figure 54: Output segmentation maps

3.3.8. Adam Optimizer

Adam is an optimization algorithm that can be used instead of the classical stochastic gradient descent procedure to update network weights iterative based in training data.

Stochastic gradient descent maintains a single learning rate for all weight updates, it does not change during training. Adam is designed to accelerate the optimization process, decrease the number of function evaluations required to reach the optima, or to improve the capability of the optimization algorithm, making use of the average of the second moments of the gradients (the uncentered variance) to adapt the learning rate. It can improve performance on problems with sparse gradients (computer vision). [17] [18]

3.3.9. Dice Loss Function

As part of the optimization algorithm, the error for the current state of the model must be estimated repeatedly. This requires the choice of loss function that can be used to estimate the loss of the model so that the weights can be updated to reduce the loss on the next evaluation. In our case, it is used Dice Loss, which originates from Sørensen–Dice coefficient, a statistic used to gauge the similarity of two samples. [19] [20]

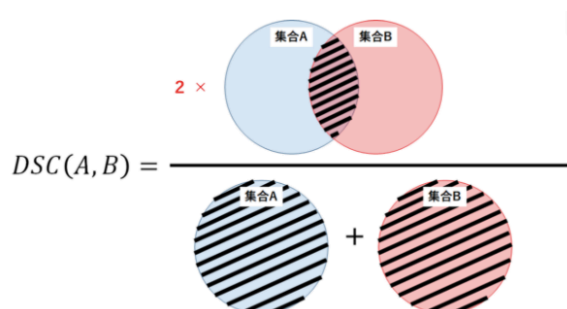


Figure 55: Sørensen–Dice coefficient [27]

So, the generic formula to compute it. **[19]**

$$DL = 1 - DSC$$

Jaccard Loss is very similar to Dice Loss but instead of $\frac{2|A \cap B|}{|A| + |B|}$ it optimizes the Jaccard Coefficient (IoU) Figure 56: IoU . MSE Loss is based on the Mean Square Error.

Where A is ground truth mask and B is predicted mask or vice versa.

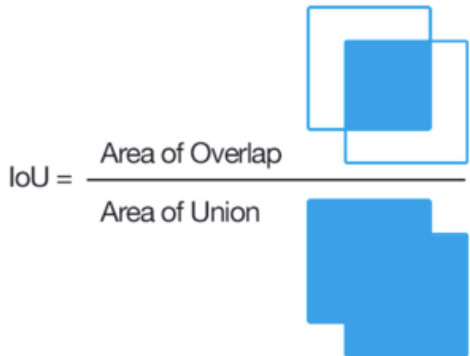
All these loss functions are applied for each mask separated, but the backpropagation is computed with the whole mini-batch.

4. Results

4.1. Metrics, IoU

The Intersection-Over-Union (IoU), also known as the Jaccard Index, is one of the most commonly used metrics in semantic segmentation.

The IoU is the area of overlap between the predicted segmentation and the ground truth divided by the area of union between the predicted segmentation and the ground truth. This metric ranges from 0–1 (0–100%) with 0 meaning no overlap and 1 meaning perfectly overlapping segmentation [22].



$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Figure 56: IoU [21]

The IoU computed in the network is calculated over the whole mini-batch, with all channels together and pixel by pixel. This means that it is not directly the averages of IoUs of each class, because the number of pixels in each class is taken into account to calculate the coefficient, so the classes with more representation counts more.

In addition, we calculate two metrics for the validation data, which both take into account all the validation images. One that calculates IoU for each class and another that determines the IoU of pixel detection, only taken into account if the pixel is detected and not the class to which it is assigned.

4.2. Final Results

To finish adjusting the results of our network, different parameterizations have been tested to see what results we obtain. We test different Loss Functions (Jaccard Loss, Dice Loss, MSE Loss), different optimizers (Adam, Adamax), different learning rates (static, variable) and different number of epochs.

Optimizer	Epochs	LR	Loss Function	Macrophage IoU	Lymphocyte IoU	Polymorph onuclear IoU	Eosinophiles IoU	Valid IoU Score	Train IoU Score	Pixel Detection IoU	
Adam	40	Fixed: 1e-3	Jacard Loss	0,72	0	0,534	0	0,61	0,623	0,68	
		Variable: 1e-3 until 30, 1e-5	Jacard Loss	0,747	0	0,598	0	0,621	0,632	0,733	
	60	Fixed: 1e-3	Dice Loss	0,747	0,473	0,531	0	0,684	0,737	0,79	
		Variable 1e-3 until 40, 1e-5 until 55, 1e-6	Dice Loss	0,741	0,46	0,544	0	0,69	0,716	0,79	
	80	Fixed 1e-3	Dice Loss	0,73	0,34	0,53	0	0,649	0,69	0,78	
		Variable 1e-3 until 50, 1e-5	Dice Loss	0,75	0,501	0,527	0	0,695	0,746	0,778	
		Variable 1e-3 until 50, 1e-5 until 70, 1e-6	Dice Loss	0,75	0,512	0,535	0	0,7	0,824	0,794	
	Adamax	40	Fixed: 1e-3	Dice Loss	0,737	0,504	0,527	0	0,619	0,691	0,774
			Variable: 1e-3 until 30, 1e-5	Jacard Loss	0,742	0,587	0,525	0	0,693	0,758	0,785
60		Variable: 1e-3 until 45, 1e-5	Dice Loss	0,748	0,501	0,527	0	0,684	0,763	0,769	
		Variable 1e-3 until 35, 1e-5 until 55, 1e-6	Dice Loss	0,74	0,587	0,536	0	0,693	0,78	0,782	
80		Variable 1e-3, 1e-5	Jacard Loss	0,745	0,562	0,581	0	0,685	0,802	0,788	
		Variable 1e-3 until 50, 1e-5 until 70, 1e-6	Dice Loss	0,75	0,498	0,53	0	0,687	0,778	0,785	

Table 8: Results Table

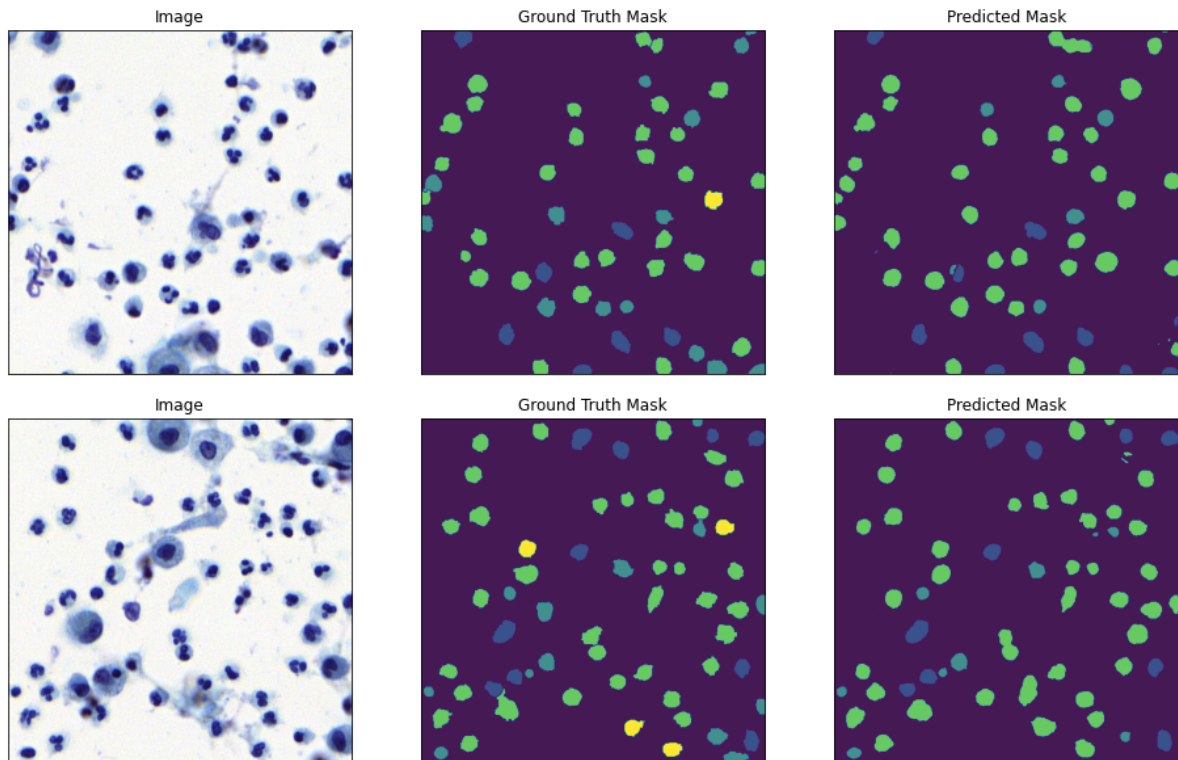


Figure 57: Validation Results

4.2.1. Loss Function

Jaccard Loss, Dice Loss and MSE Loss have been applied in the net in order to see how they work. Jaccard Loss and Dice Loss are similar, outperforming MSE Loss, they converge towards similar solutions. Looking at the results (Table 8, Table 9) Jaccard Loss seems to converge into a better solution with fewer epochs and with Adamax optimizer, whereas Dice Loss seems to converge towards a better solution with more epochs and combined with Adam optimizer.

Parametrization	Loss Function	Valid IoU Score	Train IoU Score
Adam 40 variable lr 1e-3 until 30, 1e-4	JaccardLoss	0,621	0,632
	DiceLoss	0,579	0,615
Adamax 40 variable lr 1e-3 until 30, 1e-4	JaccardLoss	0,693	0,758
	DiceLoss	0,63	0,679
Adamax 60 variable lr 1e-3 until 35, 1e-5 until 55, 1e-6	JaccardLoss	0,645	0,732
	DiceLoss	0,693	0,78
	MSELoss	0,601	0,671
Adam 80 variable lr variable lr 1e-3 until 50, 1e-5	JaccardLoss	0,686	0,723
	DiceLoss	0,695	0,746
	MSELoss	0,605	0,641
Adamax 80 variable lr variable lr 1e-3 until 50, 1e-5	JaccardLoss	0,685	0,802
	DiceLoss	0,684	0,763
	MSELoss	0,6732	0,755

Table 9: Loss Function results comparison

4.2.2. Optimizer

Adam and Adamax optimizers have been tested in the network. Adamax is an extension of Adam optimizer based on the infinity norm. Both present similar behaviours, superior to SGD. Looking at the results (Table 8, Figure 58), Adamax seems to perform better with fewer epochs and with fixed learning rates, whereas Adam seems to converge towards a better solution with more epochs and with variable learning rates. Adamax presents smoother growing in the initial validation epochs, but differs more from training results in the last epochs. Adam starts with irregular growing in the first epochs but it finally remains stable more according with training results.

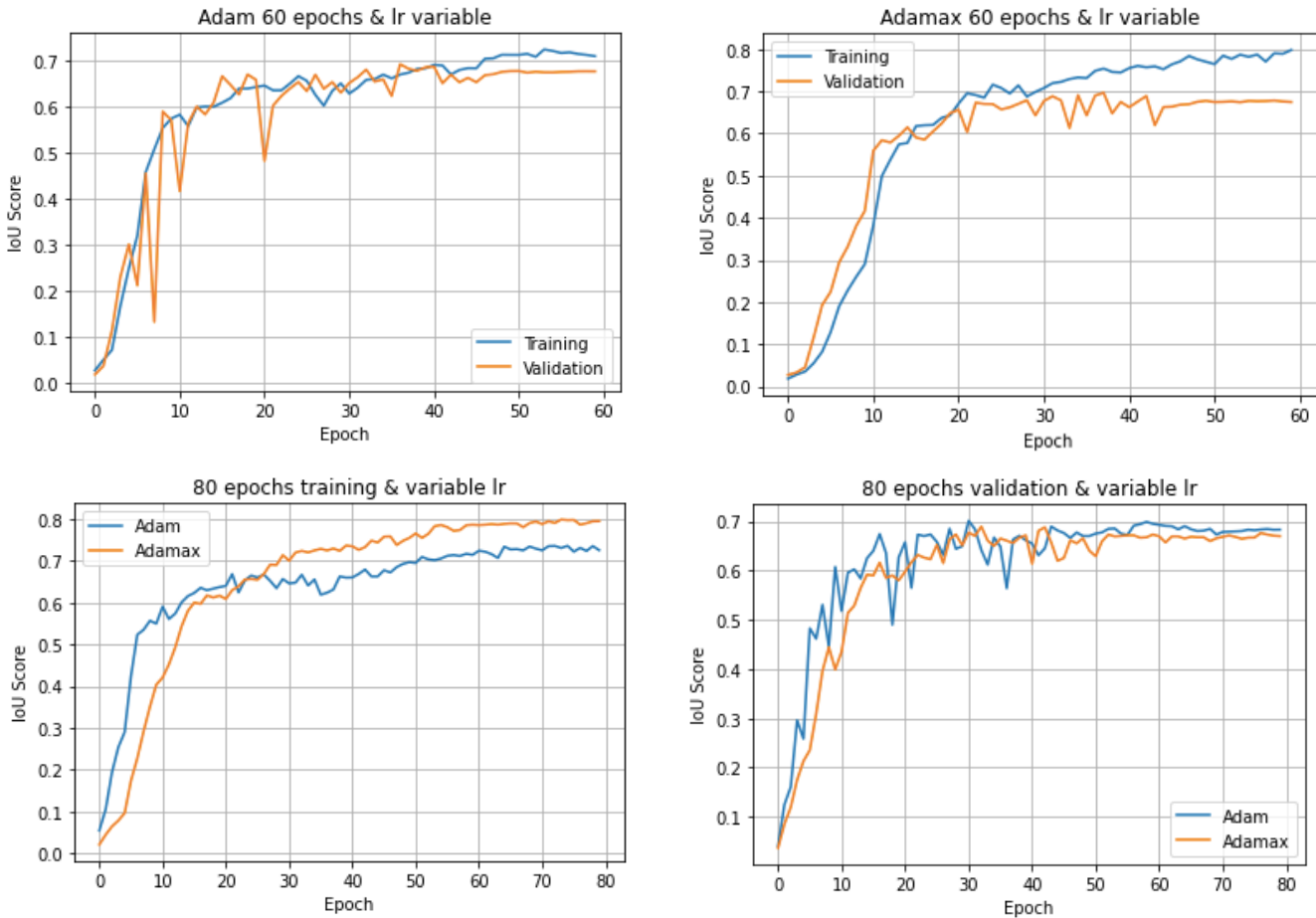


Figure 58: Adam vs Adamax

4.2.3. Learning Rate Annealing

A commonly used technique, known as learning rate annealing, recommends starting with a relatively high learning rate and then gradually lowering the learning rate during training. The intuition behind this approach is that we would like to traverse quickly from the initial parameters to a range of "good" parameter values but then we would like a learning rate small enough that we can explore the "deeper, but narrower parts of the loss function" [23].

Following this technique, we test three different learning rates: static learning rate, variable learning rate with one step, and variable learning rate with two steps. The results obtained are very similar (Table 8: Results Table, Figure 59: Static learning rate vs variable), it depends on the number of epochs we use, but varying the learning rate we achieve quickly "good" parameter values and then make a more accurate adjustment.

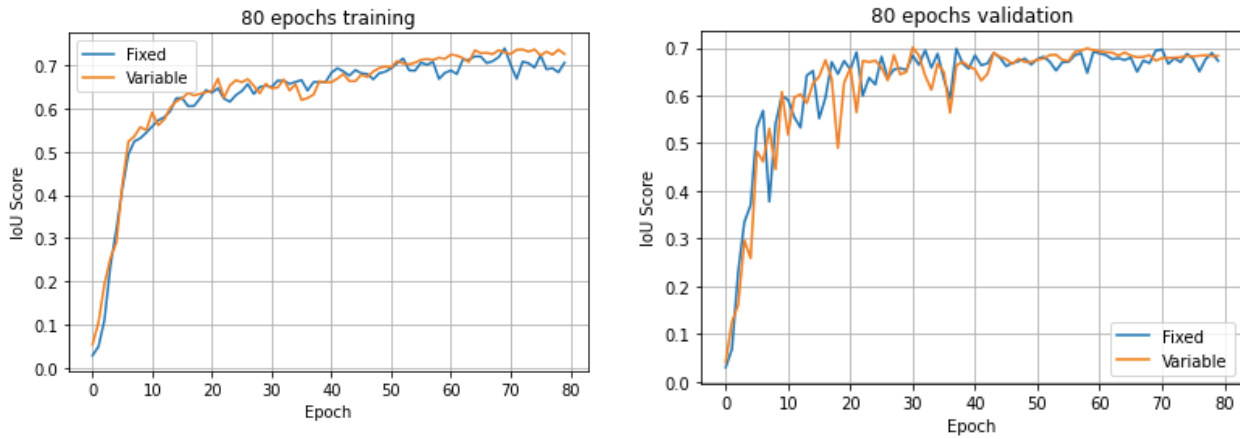


Figure 59: Static learning rate vs variable

4.2.4. Number of epochs

The number of epochs is the parameter that controls the number of complete passes through the training dataset. One of the critical issues while training a neural network on the sample data is Overfitting. When the number of epochs used to train a neural network is more than necessary, the training model learns patterns that are specific to sample data to a great extent. This makes the model incapable to perform well on a new dataset [24].

In our case, we do not obtain overfitting, our system fits the data but the validation results do not get worse along the epochs, so this is not an overfitting problem. But the model converges towards a solution and the scores almost do not vary since certain epoch, so it does not make sense to extend the number of epochs if the optimization of the weights is not going to improve.

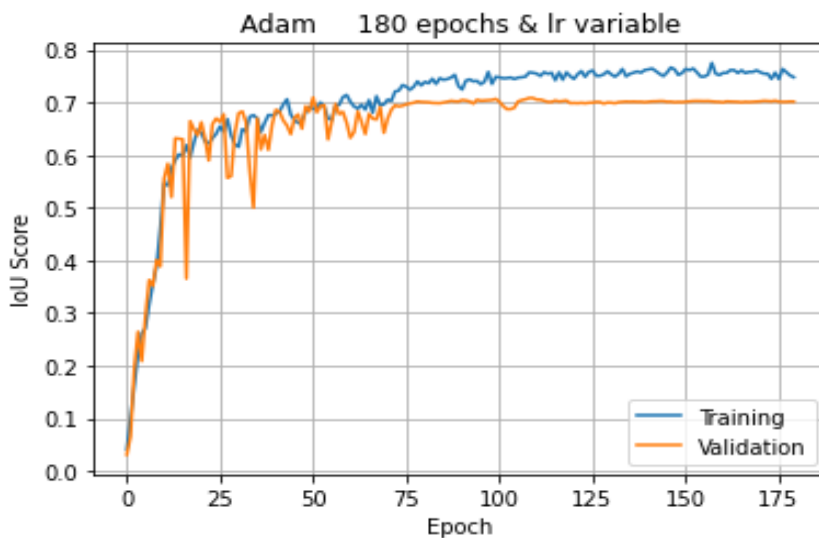


Figure 60: IoU Score along 180 epochs

40, 60 and 80 epochs have been tested with different learning rates. 40 epochs seem to be too few, despite the network starts to remain stable, bests results can be obtained. Between 60 and 80 epochs the results vary very little, they depend more on the optimizer and the variations in learning rate.

4.2.5. Global and per Class IoU Score

Looking at results (Table 8: Results Table) we can observe the global IoU score and IoU score per class computed by different network parametrizations. Global IoU score is a metric which defines how the network is working, and having scores of 70% approximately we consider them as good results in order to help for future dataset creation and development.

In case of IoU score for each class we cannot say the same, they are lower as we expected in some classes more than in others. These results make us think about the disparity in the IoU per class and the great difference in comparison with the global IoU. Knowing that the only difference between both is that the network computes the global IoU which takes into account the frequency of occurrence (at pixel level) of each class, we thought it could be a class imbalance problem.

This is a problem that plagues most of the Machine Learning / Deep Learning Classification problems. It occurs when there are one or more classes (majority classes) that are more frequent occurring than the other classes (minority classes). Simply put, there is a skewness towards the majority class [25].

To analyse if our dataset suffers of imbalance, we compute how many pixels has each class in the whole dataset, and how this affects to the IoU score per class.

	Macrophages	Lymphocytes	Polymorphonuclears	Eosinophiles
Number of pixels in training images	248988	92217	337132	18326
Number of pixels in validation images	77262	28330	83141	5409
% of the total cell pixels in training	35,74	13,24	48,39	2,63
% of the total cell pixels in validation	39,80	14,59	42,82	2,79
IoU Score of a good parametrization	0,75	0,512	0,535	0
Avg IoU Score	0,742	0,425	0,54	0

Table 10: Correlation between class representation and its IoU

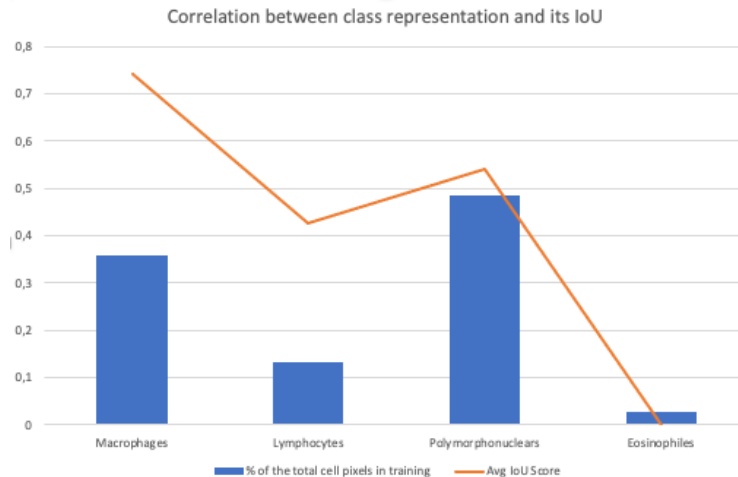


Figure 61: Correlation between class representation and its IoU

Results are clear, our dataset suffers of imbalance, the IoU score for the majority classes are clearly higher than for the minority classes (eosinophiles IoU = 0). This is because when training a model on an imbalanced dataset, the learning becomes biased towards the majority classes. With a greater number of examples available to learn from, the model learns to perform well on the majority classes, but due to the lack of enough examples, the model fails to learn meaningful patterns that could aid it in learning the minority classes [25].

Another important fact to highlight is that macrophages are the best detected cells despite not being the most representative class in the images. This is probably because it is, as we have seen previously in the maxtree, the most different class from the rest, and therefore the easiest to differentiate.

These “low” IoU per class scores makes the results of this project limited. However, their analysis can help a lot for future development, so knowing this imbalance in the dataset, actions can be taken to improve results, as the proposed in “5.2. Future Development”.

4.2.6. Pixel detection IoU score

Looking at results (Table 8: Results Table) we can observe the IoU score for the pixel detection in different parametrizations. This is one strong point in our results (80% approx.), because if we want to develop a tool to ease the creation of a big dataset, detect a high number of pixels that belong to any class we think that helps a lot. Because it provides the advantage of only have to change the label of some cells and not have to label them from zero.

4.2.7. Tricky solution

Taking into account the explanation in “4.2.5. Global and per Class IoU score”, one possible “tricky” result for the CNN, which actually can give “acceptable” IoU global scores can be obtained assigning all small cells pixels to the polymorphonuclear class. Because it was the most representative class (48,4% of the cell pixels) and because they are very similar to lymphocytes and eosinophiles. As the loss is computed globally and not for each class,

as IoU global score, it also takes into account the frequency of occurrence (at pixel level) of each class, making possible this kind of solution.

In order to avoid them in future development; something we can do is to avoid the parametrizations that lead us to these solutions. But to fix it in a good way, we should improve the dataset in order to increase the labelled data and solve the class imbalance problem.

4.2.8. Maxtree Results

Actually, results of the first part of the project were difficult to evaluate, because we did not have any labelled data to compare with, so we could not adjust or train the parameters based in a ground truth, because we did not have it. One possible solution could have been labelling manually some images to compute some scores and adjust our parameters before label the 56 images, but now it is too late, for future development I think I will implement it.

However, now that we have the small dataset, we can compute the IoU score of the previous algorithm based on maxtree.

	Macrophage IoU	Lymphocyte IoU	Polymorphonuclear IoU	Eosinophiles IoU	Detect. Pixel IoU
Best CNN Model	0,75	0,512	0,535	0	0,794
Maxtree Algorithm	0,714	0,516	0,234	0,05	0,83

Table 11: CNN vs Maxtree

As it can be seen, performance is correct; the macrophage IoU is high, the lymphocyte IoU is very close to the CNN IoU and the IoU for pixel detection is higher than the obtained by the CNN. It is true that for polymorphonuclears, a very representative class, it does not provide good results. But taking into account that the goal for this algorithm was ease the manual labelling task, we can conclude that it probably does.

One problem with this algorithm is that it was not intended to train it, so even if we improve and increase the dataset, results will not improve. So, if we want to continue facing the problem with this system, we should extract descriptors from the cuts made in maxtree to train them. But if the goal is to continue with CNN techniques, I think this maxtree based algorithm has fulfilled its objective, do as a launcher to start tackling the problem from machine learning.

5. Conclusions and future development

5.1. Conclusions

This Project has been the first contact with the BAL image segmentation. Carrying it out different techniques have been studied and applied to face this problem. As first, using a segmentation algorithm based on maxtree and correcting by hand the masks computed by it (56 masks). And finally testing and training a CNN, to develop a better classifier and obtain valuable information for the future development of the problem in CNN area.

We can conclude that, in a way, the objectives of the project have been accomplished, despite perhaps the scores are not the expected and the CNN cannot generate segmented dataset to modify only details. I think that carrying out this project we have learned and advanced a lot with the BAL segmentation. We have obtained many valuable information about our dataset, we have developed a maxtree-based algorithm that has made the masks correction easier, we have labelled and corrected 56 different images, we have tested and trained a CNN, analysed different parametrizations, studied and applied techniques to improve results in small datasets, and finally, this CNN was trained to segment new images achieving better results.

So, taking all this into account, and considering that at first, we had nothing labelled, and for training the net we have used a very limited dataset, with imbalance problems, we consider this project a good starting point.

5.2. Future Development

5.2.1. New techniques to classify lymphocytes

This was a proposed method but not tested, maybe it is useful for future development. Considering the last explanations of the doctor Jordi Temprana, who said that lymphocytes have to be very round and almost without cytoplasm, it makes sense to measure the circularity of the nuclei and the ratio nucleus / cytoplasm of the small cells and only if they are very round and their ratio is close to 1, classify the cell as lymphocyte.

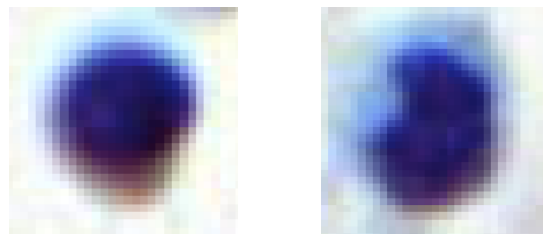


Figure 62: Lymphocyte vs Polymorphonuclear

5.2.2. Improve Transfer Learning

As you know, we used a pre-trained models to initialize our CNN, applying transfer learning. But perhaps, images used to train these previous models are not the most similar to ours. ImageNet has 1000 classes. That's why pretrained models have a lot of parameters in the

last layers on this dataset. On the other hand, medical image datasets have a small set of classes, frequently less than 20 (in our case 4). So, the design is suboptimal and probably these models are overparametrized for the medical imaging datasets [26].

In order to improve results and take more advantage of the transfer learning, find a model trained with images similar to ours would be necessary, so the initial parameters would be more adjusted to our images and our training would converge towards a better solution with less epochs by fine tuning.

5.2.3. Improve Class Imbalance

As explained, worst class results are very correlated with less data we have for this class. It seems logical, because the CNN does not receive enough data from this class so it does not learn enough to classify it.

At first, talk with doctors could be a good option, to know the roll of eosinophiles (the class with less representation in the images), the importance of their detection and if there can be tiles with high presence of them.

In case they do not have images with high presence of eosinophiles, other options can be considered.

5.2.3.1. Weighted Loss Function

In order to compensate the class imbalance, we can use a weighted loss function, which weights the loss computed for different samples differently based on the class they belong. We essentially want to assign a higher weight to the loss encountered by the samples associated with minor classes.

There are different weighting schemes that can be followed, for example INS (Inverse of Number of Samples), ISNS (Inverse of Square Root of Number of Samples, ENS (Effective Number of Samples), among others. [25]

5.2.3.2. Data Generation

Another option to compensate the dataset is generating images from clippings where eosinophiles appears. It is important to generate feasible contexts; do not increase the appearance frequency too much, accompany eosinophiles with cells that usually appear next to them... Because CNN can learn things that are not true. Also, to have more samples, more data augmentations can be applied on these images.

5.2.4. Squamous cellularity detection

Following the indications that doctors gave us, the squamous cellularity detection can be important in order to determinate pathologies in the samples. For that reason, we think in future development, the detection of these cells could be interesting. However, these cells have very little representation in the samples, so it is likely to obtain low quality detection results due to the lack of data and the imbalance of the dataset, so probably, techniques as suggested in “5.2.3. Improve Class Imbalance” should be applied.

6. Bibliography

- [1] “Wikipedia,” 21 June 2020. [Online].
Available: https://en.wikipedia.org/wiki/Bronchoalveolar_lavage.
- [2] “What is Whole Slide Imaging?,” mbf biocience, 2021. [Online].
Available: <https://www.mbfbioscience.com/whole-slide-imaging>.
- [3] J. Jordan, “An overview of semantic image segmentation.,” 21 May 2018. [Online].
Available: <https://www.jeremyjordan.me/semantic-segmentation/>.
- [4] S. L. Philippe Salembier, “Maxtree Processing Toolbox,” 29 February 2020. [Online].
Available: https://github.com/imatge-upc/Maxtree-Processing-Toolbox/blob/master/Doc/Maxtree_Processing_Toolbox_Doc.pdf.
- [5] T. G. Edwin Carlinet, “A Comparison of Many Max-tree Computation Algorithms,” 2013.
- [6] B. Kayalibay, “CNN-based Segmentation of Medical Imaging Data,” 11 January 2017.
[Online]. Available: <https://arxiv.org/abs/1701.03056>.
- [7] S. Saha, “A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way,” 15 December 2018. [Online]. Available: <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>.
- [8] S. K. Ravi Kaushik, “Image Segmentation Using Convolutional Neural Network,” *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH*, vol. 8, 2019.
- [9] P. F. a. T. B. Olaf Ronneberger, “U-Net: Convolutional Networks for Biomedical Image Segmentation,” 2015.
- [10] “Matlab `bwdist`,” [Online]. Available: <https://se.mathworks.com/help/images/ref/bwdist.html>.
- [11] A. G. a. A. Z. Abhishek Dutta, “VGG Image Annotator (VIA),” [Online].
Available: <https://www.robots.ox.ac.uk/~vgg/software/via/>.
- [12] P. Yakubovskiy, “Segmentation Models,” 2021. [Online].
Available: <https://smp.readthedocs.io/en/latest/index.html>.

- [13] A. Gandhi, "Data Augmentation | How to use Deep Learning when you have Limited Data—Part 2," 2021. [Online]. Available: <https://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/>.
- [14] V. I. I. E. K. A. P. M. D. a. A. A. K. Alexander Buslaev, "Albumentations: Fast and Flexible Image Augmentations," 2020.
- [15] JulienMaille, "github," 2021. [Online]. Available: https://github.com/qubvel/segmentation_models.pytorch#models.
- [16] L. Hulstaert, "Transfer Learning: Leverage Insights from Big Data," 19 January 2018. [Online]. Available: https://www.datacamp.com/community/tutorials/transfer-learning?utm_source=adwords_ppc&utm_campaignid=898687156&utm_adgroupid=48947256715&utm_device=c&utm_keyword=&utm_matchtype=b&utm_network=g&utm_adpostion=&utm_creative=332602034352&utm_targetid=aud-5173.
- [17] J. Brownlee, "Gentle Introduction to the Adam Optimization Algorithm for Deep Learning," 3 July 2017. [Online]. Available: <https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/>.
- [18] J. Brownlee, "Code Adam Optimization Algorithm From Scratch," 13 January 2021. [Online]. Available: <https://machinelearningmastery.com/adam-optimization-from-scratch/>.
- [19] "Sørensen–Dice coefficient," Wikipedia, 2021. [Online]. Available: https://en.wikipedia.org/wiki/S%C3%B8rensen%E2%80%93Dice_coefficient.
- [20] S. Jadon, "A survey of loss functions for semantic segmentation," 2020.
- [21] O. G. S. T. a. E. N. Joris Gu´erin, "CNN features are also great," *Laboratoire des Sciences de l'Information et des Syst`emes (CNRS UMR 7296) Arts et M´etiers ParisTech, Lille, France*, 2018.
- [22] E. Tiu, "Metrics to Evaluate your Semantic Segmentation Model," 10 August 2019. [Online]. Available: <https://towardsdatascience.com/metrics-to-evaluate-your-semantic-segmentation-model-6bcb99639aa2>.
- [23] J. JORDAN, "Setting the learning rate of your neural network.," 1 March 2018. [Online]. Available: <https://www.jeremyjordan.me/nn-learning-rate/>.
- [24] manmayi, "Choose optimal number of epochs to train a neural network in Keras," 8 June 2020. [Online]. Available: <https://www.geeksforgeeks.org/choose-optimal-number-of-epochs-to-train-a-neural-network-in-keras/>.

- [25] I. Shrivasta, "Handling Class Imbalance by Introducing Sample Weighting in the Loss Function," 17 December 2020. [Online]. Available: <https://medium.com/gumgum-tech/handling-class-imbalance-by-introducing-sample-weighting-in-the-loss-function-3bdebd8203b4>.
- [26] N. Adaloglou, "Transfer learning in medical imaging: classification and segmentation," 26 November 2020. [Online]. Available: <https://theaisummer.com/medical-imaging-transfer-learning/>.
- [27] S. Du, "Understanding Dice Loss for Crisp Boundary Detection," 2020. [Online]. Available: <https://medium.com/ai-salon/understanding-dice-loss-for-crisp-boundary-detection-bb30c2e5f62b>.
- [28] M. Toprak, "Activation Functions for Deep Learning," 14 June 2020. [Online]. Available: <https://medium.com/@toprak.mhmt/activation-functions-for-deep-learning-13d8b9b20e>.