République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université M'hamed Bougara - Boumerdès

Faculté des Sciences
Département d'Informatique

Domaine : Mathématiques Informatique
Filière : Informatique
Spécialité : Technologie de l'Information / Ingénierie du Logiciel et Traitement de l'Information

N° de l'Arrêté d'habilitation de la spécialité : arrêté n° 1351 du 09/08/2016

Mémoire de fin d'études en vu de l'obtention du Diplôme du Master Académique

Thème

La segmentation et Comptage des cellules biologiques dans les images de microscope par apprentissage profond

Présenté par :
Gharbi Aghiles
Neggazi Mohamed Lamine

Stage Pratique réalisé au :

Soutenu le 07/ 07 / 2022 Devant le jury composé de
Rahmoune Nabila : Président
Yagoubi Mohamed Riad : Examinateur
Touazi Fayçal : Encadreur

Dedication

I dedicate this work to:

To my dear mother Fatma,

And to my dear father Khelifa,

Who have stood by me every step of the way
And motivated me to achieve many goals.

To my dear sister Maria,

For her advices throughout my journey.

To all my dear friends and my partner Aghiles,

For their help and support.

To my professor Touazi and Yagoubi.

For providing all their support.

Neggazi Mohamed Lamine

I dedicate this work to:

To my dear mother Malika,

And to my dear father Mohammed,

Who have stood by me every step of the way
And motivated me to achieve many goals.

To my dear sisters Imene and Feriel and my brother Sofiane

For her advices throughout my journey.

To all my dear friends Meriem and my partner Amine,

For their help and support.

To my professor Touazi and Yagoubi.

For providing all their support.

Gharbi Aghiles

Acknowledgment

« First of all, we would like to thank God for giving us the courage and patience to accomplish this work. »

« We would like to express our great thanks to our dear parents for their moral support and encouragement. »

« We address our sincere thanks to Mr. Faycal Touazi who entrusted us with this subject and who assumed the supervision of our project, the interest he showed in our work, his benevolence, his scientific rigor, his high human qualities, constituted a precious help and allowed us to carry out this work. »

« Finally, thank you to Mr. Yagoubi and Mr. Rahmoune for accepting to be our jury, and Thanks are also addressed to all those who participated, directly or indirectly, in the development of this end-of-studies project and in particular to my family and my friends »

ABSTRACT

Cell counting is a tedious task that would benefit greatly from automation. Accurate cell counting CBC (Complete Blood Count) provides essential quantitative information and plays a key role in biological research as well as industrial and biomedical applications. Unfortunately, the commonly used manual counting method is time consuming, poorly standardized and not reproducible. The task is made even more difficult by overlapping cells and poor imaging quality. In this paper we compare between two convolutional neural networks that will segment the cells as a first phase, then count them in the second phase using 3 different algorithms (Watershed, Connected Component Labeling and Circle Hough Transform).

Keywords: CNN, Cell segmentation, Cell counting, Convolutional Neural Networks.

RÉSUMÉ

Le comptage de cellules est une tâche fastidieuse qui bénéficierait grandement de l'automatisation. Le comptage précis des cellules CBC (Complete Blood Count) fournit des informations quantitatives essentielles et joue un rôle clé dans la recherche biologique ainsi que dans les applications industrielles et biomédicales. Malheureusement, la méthode de comptage manuel couramment utilisée demande beaucoup de temps, mal standardisée et n'est pas reproductible. La tâche est rendue encore plus difficile par le chevauchement des cellules. la mauvaise qualité de l'imagerie ... , nous comparons ici deux réseaux de neurones convolutionnels qui vont segmenter les globules globules (rouges, blancs et plaquettes) comme une première phase, et les compter dans la seconde phase en utilisant 3 algorithmes (Watershed, Connected Component Labeling et Circle Hough Transform).

Mots clés : CNN, Segmentation des cellules, Comptage des cellules, Réseau de Neurones Convolutionnels.

CONTENTS

LISTE OF FIGURES

LISTE OF TABLES

INTRODUCTION

Blood carries out many vital functions as it circulates through the body. It transports oxygen from the lungs to other body tissues and carries away carbon dioxide. It carries nutrients from the digestive system to the cells of the body, and carries away wastes for excretion by the kidneys. Blood helps our body fight off infectious agents and inactivates toxins, stops bleeding through its clotting ability, and regulates our body temperature. Doctors rely on many blood tests to diagnose and monitor diseases. Some tests measure the components of blood itself; others examine substances found in the blood to identify abnormal functioning of various organs. Hence, we here propose a software system which will assist pathologists to detect blood cell count and help to find out the diseases. This information can be very helpful to a physician who, for example, is trying to identify the cause of a patient's diseases.

Earlier hematologists were performing microscopic, examination and counting of blood cells manually, which was very time-consuming and tedious process. Also, the accuracy of counting mainly depends on their expertise skill and their physical conditions. Also, because of cells complex nature, it still remains a challenging task to segment cells from its background and count them automatically.

Our work is to automate the task of cell counting, we will try to find the best solution to count red, white blood cells and platelets. Therefore, we procede in two steps:

- **The segmentation:** where we need to segment the image and remove the noise to get a clear mask on which we are going preform the counting. In this phase we used two convolutional neural networks: U-Net and SegNet.

- **The counting:** in which we will take the output mask from the first phase and apply counting algorithms on it. We used 3 algorithms to count the cells: Watershed, Connected Component Labeling, Circle Hough transform.

This thesis is presented in four chapters:

In the first chapter *Blood Cells &* present the medical background of blood cells and the importance of the complete blood count. A brief summary of artificial intelligence with all of its branches and some image processing methods is given.

In the second chapter *State Of The Art* gives an overview of the diversity of state of the art methods with a comparative study. A detailed study of the different dataset collections explored in this study is presented.

The third chapter *Contribution* present the method used in segmentation and counting.

The last chapter *Experiments And Results* gives the results of the experiments that we performed.

Chapter I : Blood cells & Machine Learning

1. Introduction

In this chapter, we will present to you the blood cells and it's three main types (red, white blood cells and platelets) as well as the complete blood count and the role of each one. Then, we will present some of the many diseases related to blood cells, and how can we detect and avoid them using artificial intelligence techniques.

In second part of this chapter, we will present machine learning with all of its branches, and then dive deeper into convolutional neural networks (CNNs) and their layers. Finally, we will present some CNN architectures and image processing methods.

2. Blood cell

2.1 Definition

A blood cell, also called a hematopoietic cell, hemocyte, or hematocyte, is a cell produced through hematopoiesis and found mainly in the blood. Major types of blood cells include red blood cells (erythrocytes), white blood cells (leukocytes), and platelets (thrombocytes). Together, these three kinds of blood cells (as we can see in fig1) add up to a total 45% of the blood tissue by volume, with the remaining 55% of the volume composed of plasma, the liquid component of blood. [17]

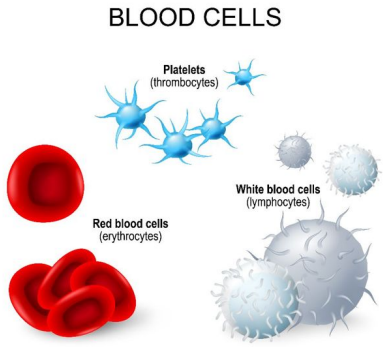


Figure 1: Blood Cell Types

That said, there are three main types of blood cells:

2.2 Red blood cells

Red blood cells (RBCs) are the cells which carry fresh oxygen all over the human body. This is vital to our health. They are round with a flattish, indented center, like doughnuts without a hole. The hemoglobin is the protein found inside the red blood cell, and it's main purpose is carrying oxygen.

Most people don't think about their red blood cells unless they have a disease that affects these cells. Problems with red blood cells can be caused by illnesses or a lack of iron or vitamins. Some diseases of the red blood cells are inherited.

Diseases of the red blood cells include many types of anemia. This is a condition in which there are too few red blood cells to carry enough oxygen all over the body.

People with anemia may have red blood cells that have an abnormal shape or that look normal, larger than normal, or smaller than normal.

Symptoms of anemia include tiredness, fast heart rate, pale skin, feeling cold, and, in severe cases, heart failure. Children who don't have enough healthy red blood cells grow and develop more slowly than other children. These symptoms show how important red blood cells are to your daily life. [40]

2.3 White blood cells

On the other hand, White blood cells (WBCs), also known as leukocytes, are the ones responsible for protecting the human body from infections. As part of the immune system, These cells circulate in the blood and respond to injury or illness. They travel through the bloodstream searching for infections. Then, they notify other WBCs of their location to help defending against attacks from other unknown organisms. Once the army of WBCs arrive, they fight the invader by producing antibody proteins to attach to the organism and destroy it. [9]

White blood cells are divided into five types:

- **Neutrophils:** Help protect your body from infections by killing bacteria, fungi and foreign debris.
- **Lymphocytes:** Consist of T cells, natural killer cells and B cells to protect against viral infections and produce proteins to help you fight infection (antibodies).

- **Eosinophils:** Identify and destroy parasites, cancer cells and assists basophils with your allergic response.
- **Basophils:** Produces an allergic response like coughing, sneezing or a runny nose.
- **Monocytes:** Defend against infection by cleaning up damaged cells.

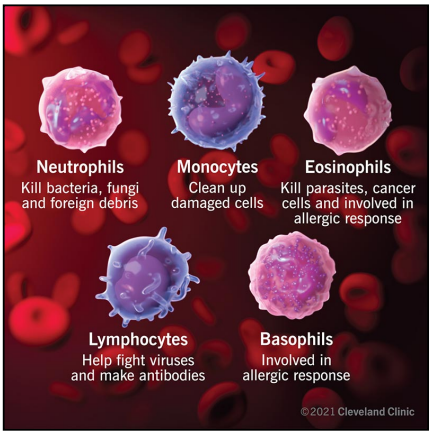


Figure 2: The five types of White blood cells

2.4 Platelets

Platelets help prevent blood loss at sites of vascular injury. To do this, they adhere, aggregate and form a procoagulant surface favorizing thrombin generation and fibrin formation. In addition, platelets express and release substances that promote tissue repair and influence processes such as angiogenesis, inflammation and the immune response. They contain large secretable pools of biologically active proteins, while newly synthesized active metabolites are also released. [37]

3. Complete blood count

3.1 Definition

A complete blood count (CBC) is a set of tests that gives information about the cells in a person's blood. A scientist or lab technician performs the requested testing and provides the requesting medical professional with the results of the CBC. In the past, counting the cells in a patient's blood was performed manually, by viewing a slide prepared with a sample of the patient's blood under a microscope. Today, this process is generally automated by use

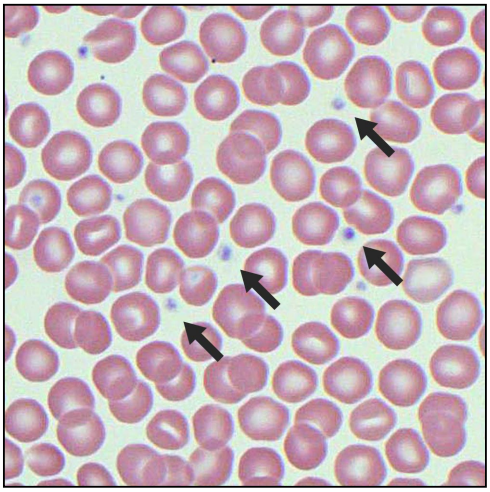


Figure 3: Platelets

of an automated analyzer, with only approximately 10-20% of samples now being examined manually. Abnormally high or low counts may indicate the presence of a disease.

3.2 Purpose of a complete blood count

A complete blood count is a common blood test that is done for a variety of reasons:

- **To review the overall health:** A doctor may recommend a complete blood count as part of a routine medical examination to monitor your general health and to screen for a variety of disorders, such as anemia or leukemia.
- **To diagnose a medical condition:** A doctor may suggest a complete blood count if you're experiencing weakness, fatigue, fever, inflammation, bruising or bleeding. A complete blood count may help diagnose the cause of these signs and symptoms. If a doctor suspects someone with an infection, the CBC test can help confirm that diagnosis.
- **To monitor a medical condition:** If you've been diagnosed with a blood disorder that affects blood cell counts, the doctor may use complete blood counts to monitor your condition.
- **To monitor medical treatment:** A complete blood count may be used to monitor your health if you're taking medications that may affect blood cell counts. [31]

3.3 Diseases identified by a complete blood count

Abnormal test results could be an indication of a serious health problem or a simple one that can be remedied by eating better or taking supplements.
These are some of the health problems that can be identified by a CBC:

- Anemia
- Autoimmune disorders
- Bone marrow problems
- Leukemia (Cancer)
- Dehydration
- Heart disease
- Infection
- Inflammation
- Vitamin and mineral deficiencies

4. Machine Learning

4.1 Definition

It is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable. [32]
Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. [19]

Machine Learning algorithms can be categorised into these four main types:

- **Supervised learning:** Supervised learning, as the name indicates, has the presence of a supervisor as a teacher. Basically supervised learning is when we teach or train the

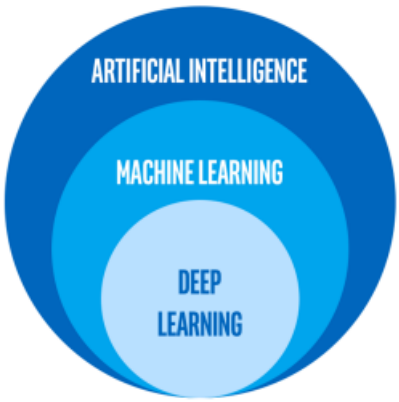


Figure 4: Artificial Intelligence

machine using data that is well labeled. Which means some data is already tagged with the correct answer. After that, the machine is provided with a new set of examples (data) so that the supervised learning algorithm analyses the training data (set of training examples) and produces a correct outcome from labeled data. [14]

- **Unsupervised learning:** Unsupervised learning is the training of a machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of the machine is to group unsorted information according to similarities, patterns, and differences without any prior training of data. [14]

Parameters	Supervised machine learning	Unsupervised machine learning
Input Data	Algorithms are trained using labeled data.	Algorithms are used against data that is not labeled
Computational Complexity	Simpler method	Computationally complex
Accuracy	Highly accurate	Less accurate

Table 1: Table that presents diffrence between supervised and unsupervised machine learning [14]

- **Semi Supervised learning:** Semi-supervised learning sits somewhere between Supervised and Unsupervised learning algorithms. It employs a mix of labeled and unlabeled datasets. It works with data that has only a few labels; it usually works with unlabeled data. Labels are expensive, yet for corporate purposes, a few labels may suffice.

- **Reinforcement learning:** Reinforcement learning is just a machine learning approach that rewards positive behavior while penalizing poor behavior. In general, a reinforcement learning agent is capable of sensing and interpreting its environment, acting, and learning via trial and error. Developers of reinforcement learning propose a way of rewarding desired behaviors and punishing negative behaviors. [2]

Parameters	Semi-Supervised Learning	Reinforcement Learning
Definition	Uses a small amount of labeled data bolstering a larger set of unlabeled data	An algorithm witha reward system
Aim	To counter the disadvantages of supervised and unsupervised learning.	To learn a series of action
Interaction of the agent	Doesn't interact	Interacts
Practical application	Speech analysis, internet content classification	Trajectory optimization, motion planning
Labels	It has labels.	It doesn't have labels.

Table 2: Comparison Table Between Semi-Supervised and Reinforcement Learning [2]

4.2 Machine Learning Models

- **Linear Regression:** Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables they are considering, and the number of independent variables getting used. [28]

- **Support Vector Machines:** Support Vector Machine (SVM) is a computer algorithm that learns by example to assign labels to objects. [6] For instance, an SVM can learn to recognize fraudulent credit card activity by examining hundreds or thousands of fraudulent and non fraudulent credit card activity reports. Alternatively, an SVM can

learn to recognize handwritten digits by examining a large collection of scanned images of handwritten zeroes, ones and so fourth. [36]

- **Gradient Descent:** Gradient descent is an optimization algorithm used to minimize some function by iteratively moving in the direction of steepest descent as defined by the negative of the gradient. In machine learning, we use gradient descent to update the parameters of our model. Parameters refer to coefficients in Linear Regression and weights in neural networks. [34]

4.3 Deep Learning

Neural Networks

Artificial neural networks (ANNs) are comprised of a node layers, containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.

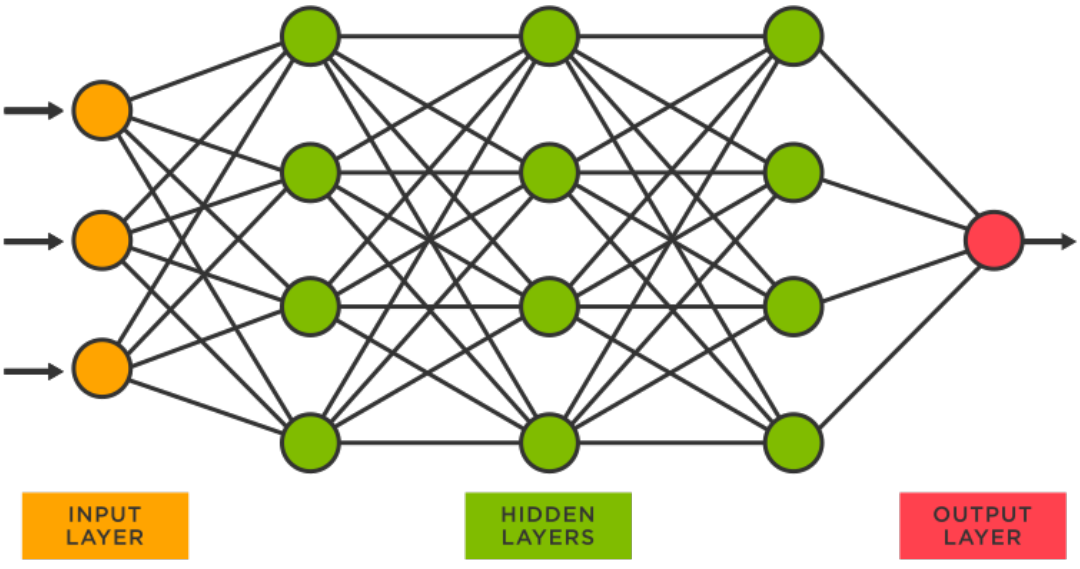


Figure 5: Architecture of Neural Network

Neural networks rely on training data to learn and improve their accuracy over time. However, once these learning algorithms are fine-tuned for accuracy, they are powerful tools in computer science and artificial intelligence, allowing us to classify and cluster data

at a high velocity. Tasks in speech recognition or image recognition can take minutes versus hours when compared to the manual identification by human experts. One of the most well-known neural networks is Google's search algorithm.

if we dive into the details, we can consider that each node has it's linear regression model, composed of input data, weights, a bias (or threshold), and an output. The formula would look something like equation 1:

$$\sum_{i=1}^{m} W_i X_i + bias = W_1 X_1 + W_2 X_2 + W_3 X_3 + bias \tag{1}$$

$$output = f(x) = \begin{cases} 1 & if \sum_{i=1}^{m} W_1 X_1 + b \leq 0 \\ 0 & if \sum_{i=1}^{m} W_1 X_1 + b < 0 \end{cases} \tag{2}$$

Once an input layer is determined, weights are assigned. These weights help determine the importance of any given variable, with larger ones contributing more significantly to the output compared to other inputs. All inputs are then multiplied by their respective weights and then summed (similar to equation 1). Afterward, the output is passed through an activation function as we can see in equation 2, which determines the output. If that output exceeds a given threshold, it "fires" (or activates) the node, passing data to the next layer in the network. This results in the output of one node becoming in the input of the next node. This process of passing data from one layer to the next layer defines this neural network as a feed-forward network.

Convolutional Neural Networks

CNNs or ConvNets are among the most successful and widely used architectures in the deep learning community, especially for computer vision tasks. CNNs were initially proposed by Fukushima in his seminal paper on the "Neocognitron" [13].

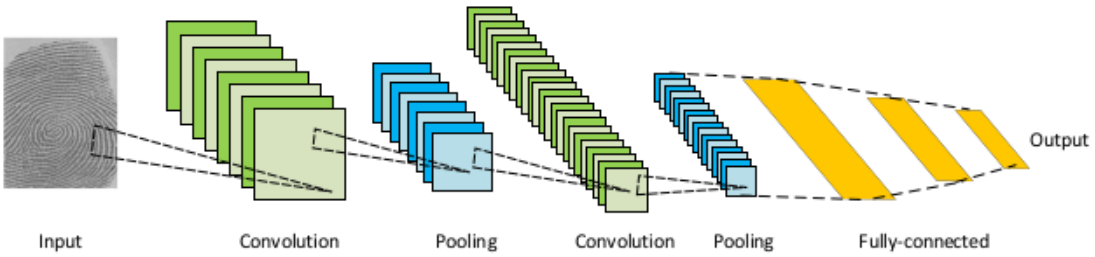


Figure 6: Architecture of convolutional neural networks. From [33]

Convolutional neural networks are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs. They have three main types of layers, which are:

- Convolutional layer
- Pooling layer
- Fully-connected (FC) layer

The convolutional layer is the first layer of a convolutional network. While convolutional layers can chained by additional convolutional layers or pooling layers, the fully-connected layer is the final layer. With each layer, the CNN increases in its complexity, identifying greater portions of the image. Earlier layers focus on simple features, such as colors and edges. As the image data progresses through the layers of the CNN, it starts to recognize larger elements or shapes of the object until it finally identifies the intended object.

1. **Convolutional layer** :

   The convolutional layer is the core building block of a CNN, and it is where the majority of computation occurs. It requires a few components, which are input data, a filter, and will output a feature map. Let's assume that the input will be a color image, which is made up of a matrix of pixels in 3D. This means that the input will have three dimensions—a height, width, and depth—which correspond to RGB in an image. We also have a feature detector, also known as a kernel or a filter, which will move across the receptive fields of the image, checking if the feature is present. This process is known as a convolution. [18]

The feature detector is a two-dimensional (2-D) array of weights, which represents part of the image. While they can vary in size, the filter size is typically a 3x3 matrix; this also determines the size of the receptive field. The filter is then applied to an area of the image, and a dot product is calculated between the input pixels and the filter. This dot product is then fed into an output array. Afterwards, the filter shifts by a stride, repeating the process until the kernel has swept across the entire image. The final output from the series of dot products from the input and the filter is known as a feature map, activation map, or a convolved feature.

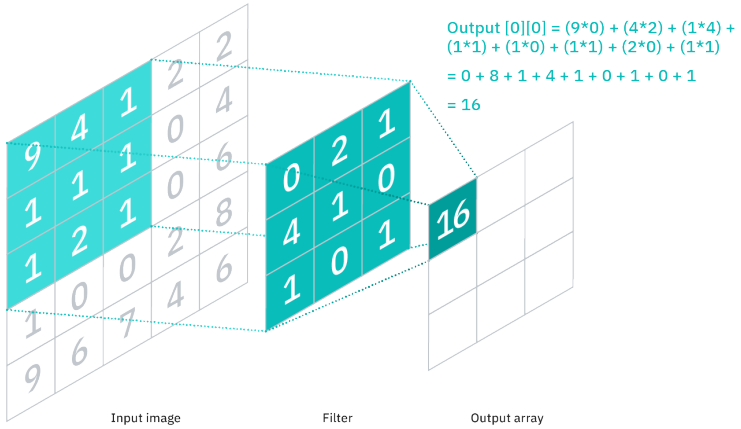


Figure 7: CNN kernel

As we can see in the fig 7, the kernel will browse all the matrix by shifting it's position. where the weights in the kernel will remain fixed as it moves across the image, which is also known as parameter sharing. Some parameters, like the weight values, adjust during training through the process of backpropagation and gradient descent. However, there are three hyperparameters which affect the volume size of the output that need to be set before the training of the neural network begins. These include:

- **The number of filters** affects the depth of the output. For example, three distinct filters will give us three different feature maps, creating a depth of three.
- **Stride** is the distance, or number of pixels, that the kernel moves over the input matrix. While stride values of two or greater is rare, a larger stride yields a smaller output.
- **Zero-padding** is usually used when the filters do not fit the input image. This

sets all elements that fall outside of the input matrix to zero, producing a larger or equally sized output. There are three types of padding:

- **Valid padding**: This is also known as no padding. In this case, the last convolution is dropped if dimensions do not align.
- **Same padding**: This padding ensures that the output layer has the same size as the input layer
- **Full padding**: This type of padding increases the size of the output by adding zeros to the border of the input.

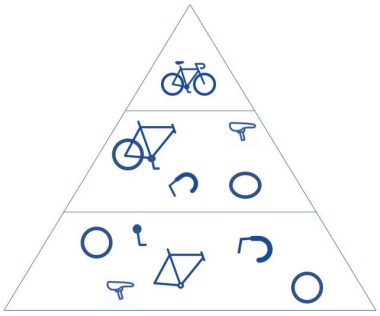


Figure 8: Feature Hierarchy

After each convolution operation, a CNN applies an activation function to the feature map, the most used activation function is the Rectified Linear Unit (ReLU).

As we mentioned earlier, when we chain convolutional layers, the structure of the CNN can become hierarchical as the later layers can see the pixels within the receptive fields of prior layers. As an example, let's assume that we're trying to determine if an image contains a bicycle. You can think of the bicycle as a sum of parts. It is comprised of a frame, handlebars, wheels, pedals, and cetera. Each individual part of the bicycle makes up a lower-level pattern in the neural net, and the combination of its parts represents a higher-level pattern, creating a feature hierarchy within the CNN.

2. **Pooling Layer**:

   Pooling layers, also known as down-sampling, conducts dimensionality reduction, reducing the number of parameters in the input. Similar to the convolutional layer, the pooling operation sweeps a filter across the entire input, but the difference is that this

filter does not have any weights. Instead, the kernel applies an aggregation function to the values within the receptive field, populating the output array. There are two main types of pooling:

- **Max pooling**: As the filter moves across the input, it selects the pixel with the maximum value to send to the output array. As an aside, this approach tends to be used more often compared to average pooling.
- **Average pooling**: As the filter moves across the input, it calculates the average value within the receptive field to send to the output array.

While a lot of information is lost in the pooling layer, it also has a number of benefits to the CNN. They help to reduce complexity, improve efficiency, and limit risk of over-fitting.

3. **Fully-Connected Layer**:
   The name of the fully-connected layer aptly describes itself. The pixel values of the input image are not directly connected to the output layer in partially connected layers. However, in the fully-connected layer, each node in the output layer connects directly to a node in the previous layer.

   This layer performs the task of classification based on the features extracted through the previous layers and their different filters. While convolutional and pooling layers tend to use ReLu functions or other activation functions, fully connected layers usually leverage a softmax or sigmoid activation function to classify inputs appropriately, producing a probability from 0 to 1.

5. CNN architectures

In this section we are going to present some CNN architectures, CNN architectures are pre-Designed models targeting a spcial task like image segmentation, feature extraction, classification ..., all the architectures can be an encoder or decoder and it can be both.

- **encoder:** also called contracting path which consist of the down-sampling of the feature map by using multiple convolutional and max pooling layers and it differs between model architectures.

- **decoder:** also called expansive path which consist of an up-sampling of the feature map by using up-convolution (which concatenates the feature map from the upper parallel encoder layers)

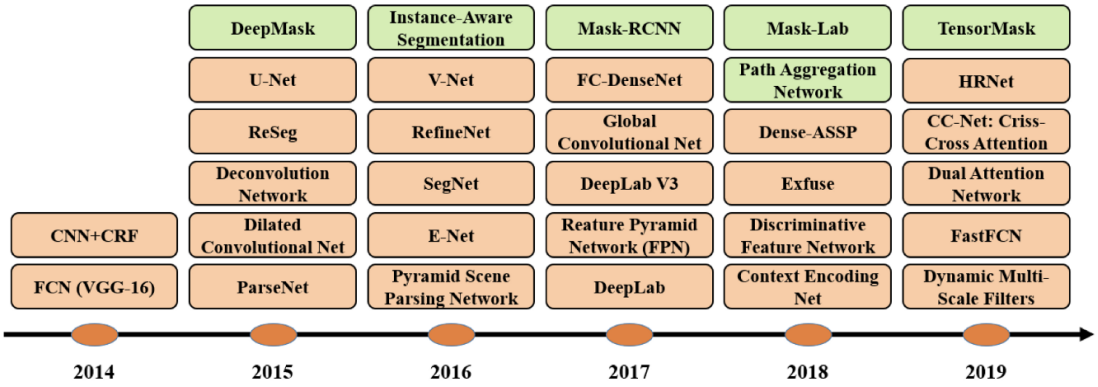


Figure 9: Timeline of segmentation algorithms

5.1 VGG

VGG stands for Visual Geometry Group; it is a standard deep Convolutional Neural Network (CNN) architecture with multiple layers. The "deep" refers to the number of layers with VGG-16 or VGG-19 consisting of 16 and 19 convolutional layers.

The VGG architecture is the basis of ground-breaking object recognition models. Developed as a deep neural network, the VGGNet also surpasses baselines on many tasks and datasets beyond ImageNet. Moreover, it is now still one of the most popular image recognition architectures.

VGG16

The VGG model, or VGGNet, that supports 16 layers is also referred to as VGG16, which is a convolutional neural network model proposed by A. Zisserman and K. Simonyan from the University of Oxford in 2014. These researchers published their model in the research paper titled, "Very Deep Convolutional Networks for Large-Scale Image Recognition." [46]

The VGG16 model achieves almost 92.7% top-5 test accuracy in ImageNet. ImageNet is a dataset consisting of more than 14 million images belonging to nearly 1000 classes.

Moreover, it was one of the most popular models submitted to ILSVRC-2014. It replaces the large kernel-sized filters with several 3×3 kernel-sized filters one after the other, thereby making significant improvements over AlexNet. The VGG16 model was trained using Nvidia Titan Black GPUs for multiple weeks.

As mentioned above, the VGGNet-16 supports 16 layers and can classify images into 1000 object categories, including keyboard, animals, pencil, mouse, etc. Additionally, the model has an image input size of 224-by-224.

VGG19

The concept of the VGG19 model (also VGGNet-19) is the same as the VGG16 except that it supports 19 layers. The "16" and "19" stand for the number of weight layers in the model (convolutional layers). This means that VGG19 has three more convolutional layers than VGG16.

VGG Architecture

VGGNets are based on the most essential features of convolutional neural networks (CNN). The VGG network is constructed with very small convolutional filters. The VGG-16 consists of 13 convolutional layers and three fully connected layers. To sum it all up, VGG architecture consists of:

- **Input:** The VGGNet takes in an image input size of 224×224. For the ImageNet competition, the creators of the model cropped out the center 224×224 patch in each image to keep the input size of the image consistent.
- **Convolutional Layers:** VGG's convolutional layers leverage a minimal receptive field, i.e., 3×3, the smallest possible size that still captures up/down and left/right. Moreover, there are also 1×1 convolution filters acting as a linear transformation of the input. This is followed by a ReLU unit, which is a huge innovation from AlexNet that reduces training time. ReLU stands for rectified linear unit activation function; it is a piece-wise linear function that will output the input if positive; otherwise, the output is zero. The convolution stride is fixed at 1 pixel to keep the spatial resolution preserved after convolution (stride is the number of pixel shifts over the input matrix).

- **Hidden Layers:** All the hidden layers in the VGG network use ReLU. VGG does not usually leverage Local Response Normalization (LRN) as it increases memory consumption and training time. Moreover, it makes no improvements to overall accuracy.

- **Fully-Connected Layers:** The VGGNet has three fully connected layers. Out of the three layers, the first two have 4096 channels each, and the third has 1000 channels, 1 for each class.

VGG16 architecture

The number 16 in the name VGG refers to the fact that it is 16 layers deep neural network (VGGnet). This means that VGG16 is a pretty extensive network and has a total of around 138 million parameters. Even according to modern standards, it is a huge network. However, VGGNet16 architecture's simplicity is what makes the network more appealing. Just by looking at its architecture, it can be said that it is quite uniform.

There are a few convolution layers followed by a pooling layer that reduces the height and the width. If we look at the number of filters that we can use, around 64 filters are available that we can double to about 128 and then to 256 filters. In the last layers, we can use 512 filters.

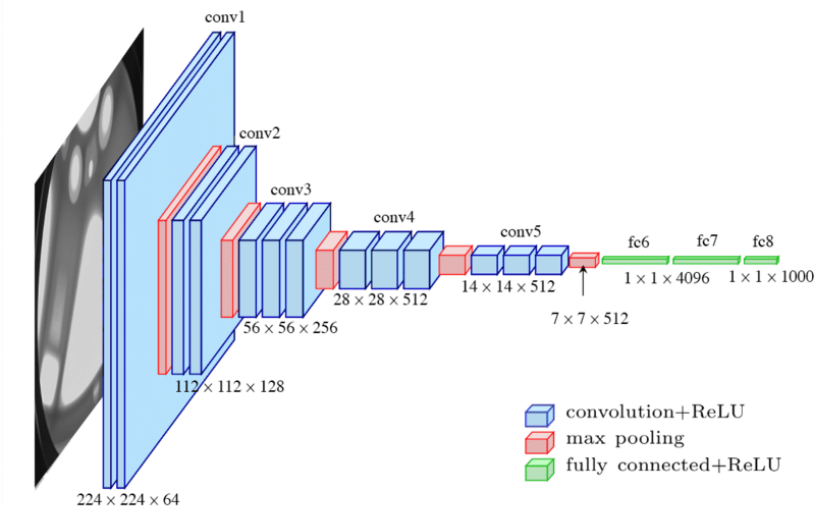


Figure 10: VGG16 architecture

Performance of VGG Models

VGG16 highly surpasses the previous versions of models in the ILSVRC-2012 and ILSVRC-2013 competitions. Moreover, the VGG16 result is competing for the classification task winner (GoogLeNet with 6.7% error) and considerably outperforms the ILSVRC-2013 winning submission Clarifai. It obtained 11.2% with external training data and around 11.7% without it. In terms of the single-net performance, the VGGNet-16 model achieves the best result with about 7.0% test error, thereby surpassing a single GoogLeNet by around 0.9%. [5]

5.2 U-Net

U-Net is a CNN architecture built for biomedical image segmentation in 2015 [41]. It consists of a contracting path (encoder) and an expansive path (decoder). The encoder follows the typical architecture of a convolutional network. It consists of the repeated application of two 3x3 convolutions (unpadded convolutions), each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride 2 for down-sampling. At each down-sampling step we double the number of feature channels. Every step in the expansive path consists of an up-sampling 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 a ReLU. The cropping is necessary due to the loss of border pixels in every convolution. At the final layer a 1x1 convolution is used to map each 64-component feature vector to the desired number of classes. In total the network has 23 convolutional layers [26].

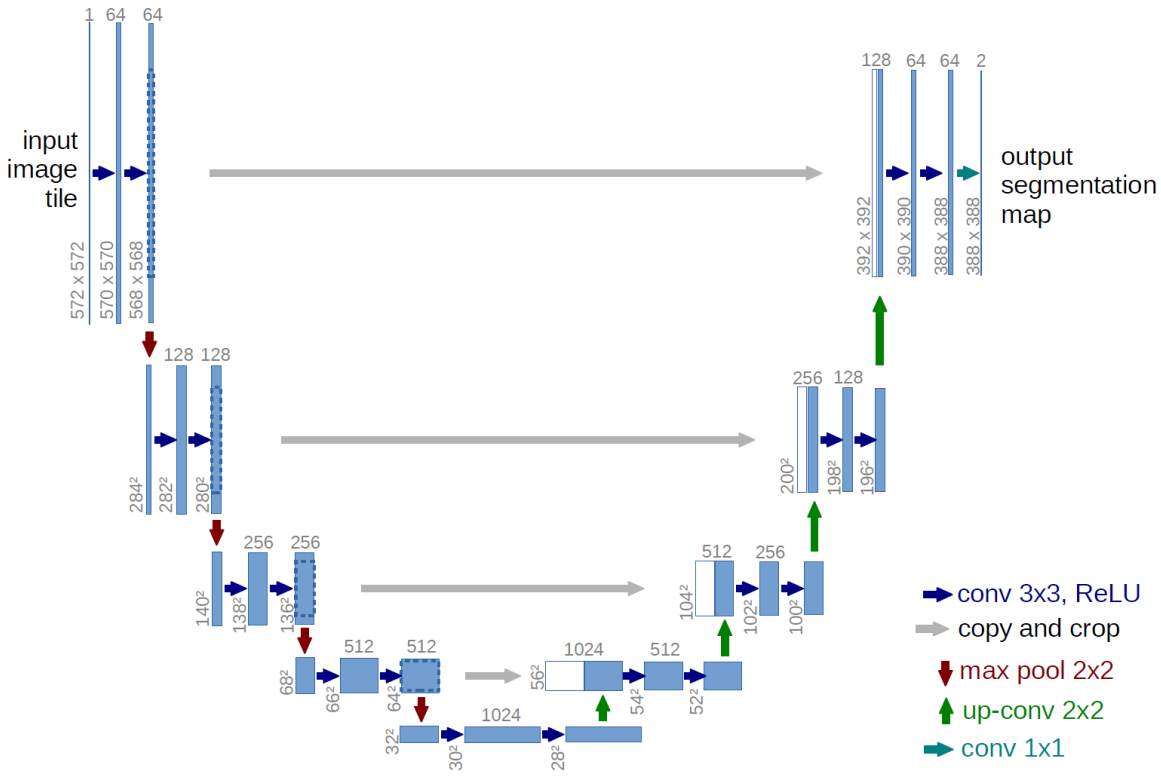


Figure 11: U-Net Architecture. From [41]

Performance

The u-net is fast and precise model for segmentation of images. Up to now it has outperformed the prior best method (a sliding-window convolutional network) on the ISBI challenge for segmentation of neuronal structures in electron microscopic stacks. It has won the Grand Challenge for Computer-Automated Detection of Caries in Bitewing Radiography at ISBI 2015, and it has won the Cell Tracking Challenge at ISBI 2015 on the two most challenging transmitted light microscopy categories (Phase contrast and DIC microscopy) by a large margin.

5.3 SegNet

SegNet is a semantic segmentation model designed in 2017. This core trainable segmentation architecture consists of an encoder network, a corresponding decoder network followed by a pixel-wise classification layer. The architecture of the encoder network is topologically identical to the 13 convolutional layers in the VGG16 network. The role of the decoder network is to map the low resolution encoder feature maps to full input resolution feature maps for pixel-wise classification. The novelty of SegNet lies is in the

manner in which the decoder up-samples its lower resolution input feature maps. Specifically, the decoder uses pooling indices computed in the max-pooling step of the corresponding encoder to perform non-linear up-sampling. [3]

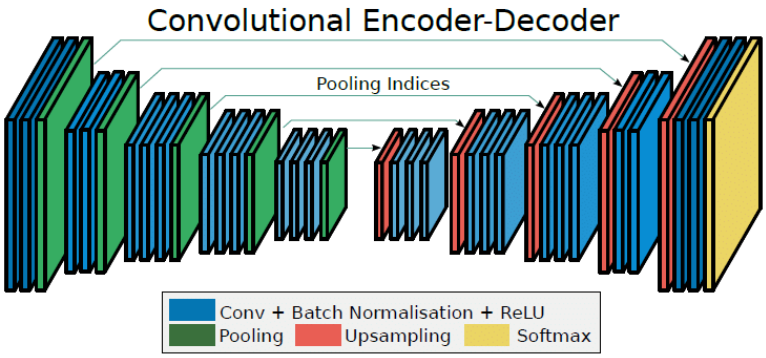
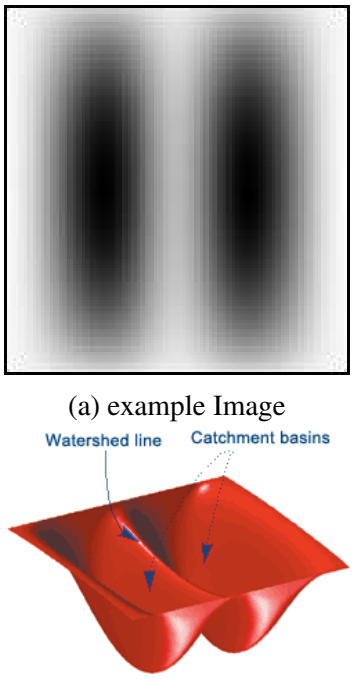


Figure 12: SegNet architecture

6. Classic Methods

6.1 Watershed

Watershed segmentation is a region-based method that has its origins in mathematical morphology, The general concept was introduced by [10]. In watershed segmentation an image is regarded as a topographic landscape with ridges and valleys. The elevation values of the landscape are typically defined by the gray values of the respective pixels or their gradient magnitude. Based on such a 3D representation the watershed transform decomposes an image into catchment basins. For each local minimum, a catchment basin comprises all points whose path of steepest descent terminates at this minimum. Watersheds separate basins from each other. The watershed transform decomposes an image completely and thus assigns each pixel either to a region or a watershed. With noisy medical image data, a large number of small regions arises. This is known as the "over-segmentation" problem. [50]



(a) example Image

(b) watershed reliefs

Figure 13: Watershed algorithm from [11]

In any grayscale images, there are areas where the intensity is high and there are some where intensity is low. We can denote these high intensity areas to peaks while low intensity areas to valleys. We think of an image as a topography map, where each pixel's intensity contributes to the topography map, either a local elevation or a depression. To separate

objects in images, we will fill out each valley with water of different colors. Slowly, the water will rise up and to a point water from different valleys start to merge. This is when we build barriers on top of the peak to avoid having the peak underwater. Once the barriers are built out, the barriers constitutes the boundary of the object.

In our case we will use the watershed by Flooding + predefined water sources. We define euclidean distance transform map as our relief map, then extract local maxima from the euclidean distance map to use them as water sources.

you can consult the **chapter 3 - section 7.3** for the use case of watershed algorithm in our system.

6.2 Connected Component Labeling

Connected component labeling (also known as connected component analysis, blob extraction, or region labeling) is an algorithmic application of graph theory used to determine the connectivity of "blob"-like regions in a binary image.

When using contour analysis, we are often restricted by the hierarchy of the outlines (i.e., one contour contained within another). With connected component analysis, we can more easily segment and analyze these structures.

A great example of connected component Labeling is that we are using this algorithm in cell counting, when we get the image mask we are using it to label all the non connected objects of the image, each object is labeled with a unique color as we can see in fig 14.

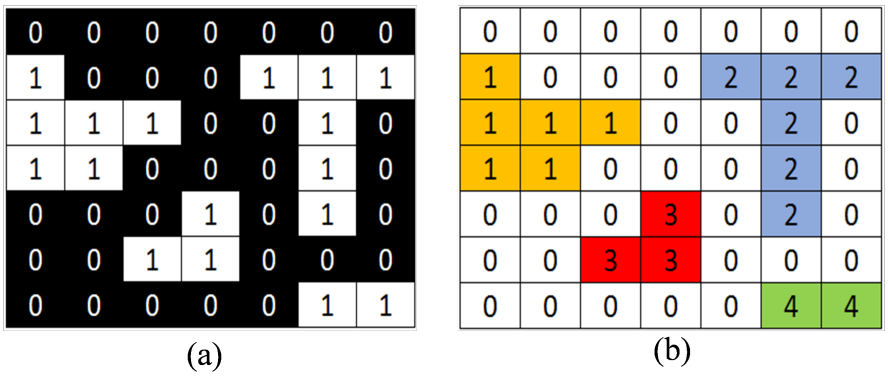


Figure 14: Object counting using Connected Component Labeling (CCL).

In our case we are using the two pass method where the algorithm will browse the matrix 2 times to create the segments which also known as the Hoshen–Kopelman algorithm

[29]. it iterates through 2-dimensional binary data. The algorithm makes two passes over the image: the first pass to assign temporary labels and record equivalences, and the second pass to replace each temporary label by the smallest label of its equivalence class.

Connectivity checks are carried out by checking neighbor pixels' labels (neighbor elements whose labels are not assigned yet are ignored), or say, the North-East, the North, the North-West and the West of the current pixel (assuming 8-connectivity). 4-connectivity uses only North and West neighbors of the current pixel. The following conditions are checked to determine the value of the label to be assigned to the current pixel (4-connectivity is assumed)

1. Does the pixel to the left (West) have the same value as the current pixel?
    - **Yes** – We are in the same region. Assign the same label to the current pixel
    - **No** – Check next condition
2. Do both pixels to the North and West of the current pixel have the same value as the current pixel but not the same label?
    - **Yes** – We know that the North and West pixels belong to the same region and must be merged. Assign the current pixel the minimum of the North and West labels, and record their equivalence relationship
    - **No** – Check next condition
3. Does the pixel to the left (West) have a different value and the one to the North the same value as the current pixel?
    - **Yes** – Assign the label of the North pixel to the current pixel
    - **No** – Check next condition
4. Do the pixel's North and West neighbors have different pixel values than current pixel?
    - **Yes** – Create a new label id and assign it to the current pixel

You can consult **Chapter 3 - Section 7.2** for the use case of CCL in our System.

6.3 Surface Filter

In this project we created our proper method for removing noise and classify objects which relies on the connected component labeling algorithm, this method is used to filter non connected objects in a binary image by surface, where it takes the minimum and maximum surface in pixels, and returns a clean image containing only the objects that satisfies the surface condition.

1. binarize the image by applying a threshold.
2. Apply Connected Component Labeling on the image to label each non connected region.
3. calculate the surface of each region segmented with CCL then apply the conditions.
4. remove all the regions that doesn't satisfy the min or max condition.

6.4 Discrete cosine transform

A discrete cosine transform (DCT) expresses a finite sequence of data points in terms of a sum of cosine functions oscillating at different frequencies. The DCT, first proposed by Nasir Ahmed in 1972 [1], is a widely used transformation technique in signal processing and data compression. It is used in most digital media, including digital images, digital video, digital audio, digital television, digital radio, and speech coding. DCTs are also important to numerous other applications in science and engineering, such as digital signal processing, telecommunication devices, reducing network bandwidth usage, and spectral methods for the numerical solution of partial differential equations. the equation for the discrete cosine transform is represented as:

$$c(u,v) = \alpha(u)\alpha(v) \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} f(x,y) cos\left[\frac{\pi(2x+1)u}{2N}\right] cos\left[\frac{\pi(2y+1)v}{2N}\right], \tag{3}$$

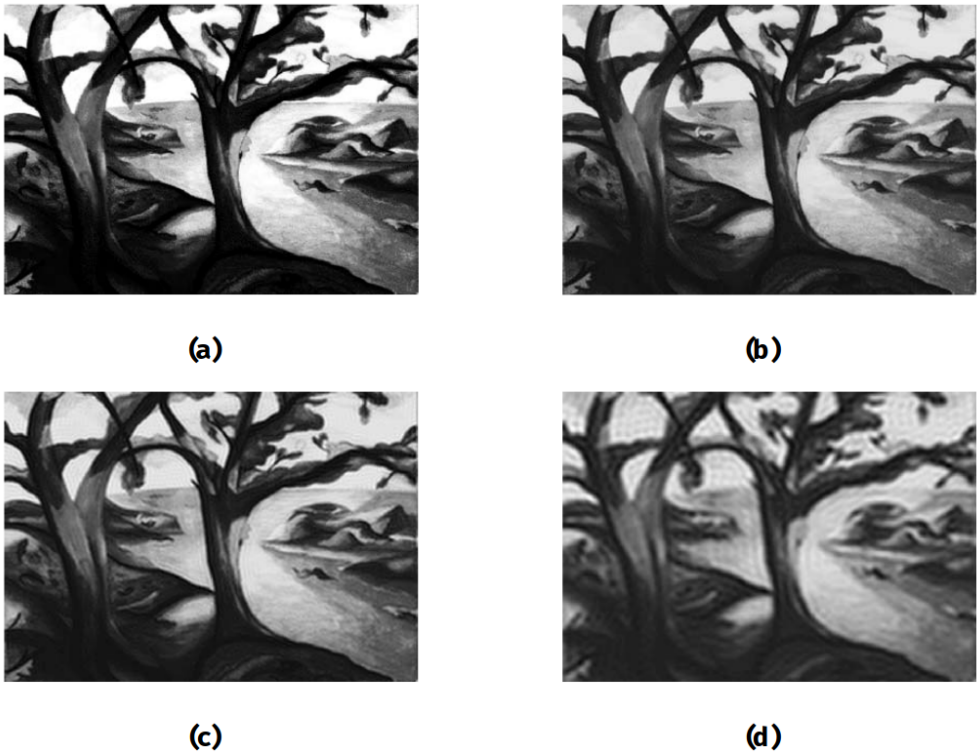


Figure 15: Inverse DCT of Trees; (a) DCT(100%); (b) DCT(75%); (c) DCT(50%); (d) DCT(25%).

6.5 Circular Hough Transform

The general Hough transform can be used on any kind of shape, although the complexity of the transformation increase with the number of parameters needed to describing the shape. [39] The Circle Hough Transform (CHT) is a basic feature extraction technique used in digital image processing for detecting circles in imperfect images.

In a two-dimensional space, a circle can be described by:

$$(x-a)^2 + (y-b)^2 = r^2 \tag{4}$$

where (a,b) is the center of the circle, and r is the radius. If a 2D point (x,y) is fixed, then the parameters can be found according to (1). The parameter space would be three dimensional, (a, b, r). And all the parameters that satisfy (x, y) would lie on the surface of an inverted right-angled cone whose apex is at (x, y, 0). In the 3D space, the circle parameters can be identified by the intersection of many conic surfaces that are defined by points on the 2D circle. This process can be divided into two stages. The first stage is fixing radius then find the optimal center of circles in a 2D parameter space. The second stage is to find the

optimal radius in a one dimensional parameter space.
This is the CHT algorithm:

1. For each A[a,b,r] = 0: where the radius is in a given min-max range.
2. Process the filtering algorithm:
   (a) apply image Gaussian Blurring.
   (b) convert the image to grayscale.
   (c) apply Canny Algorithm to extract edges of all non connected objects in the image.
3. For each region edge, Vote the all possible circles in accumulator.
4. The local maximum voted circles of Accumulator A gives the circle Hough space.
5. The maximum voted circle of Accumulator gives the circle. [52]

We can note that for each given radius another accumulator should be used.
You can consult **Chapter 3 - Section 7.1** for the use case of CHT in our System.

6.6 Distance Transform

A Distance Transform, also known as distance map or distance field, is a derived representation of a digital image. The choice of the term depends on the point of view on the object in question: whether the initial image is transformed into another representation, or it is simply endowed with an additional map or field.

Distance fields can also be signed, in the case where it is important to distinguish whether the point is inside or outside of the shape. [12]

The map labels each pixel of the image with the distance to the nearest obstacle pixel. A most common type of obstacle pixel is a boundary pixel in a binary image.

6.7 Marching Squares Algorithm

Marching squares is a computer graphics algorithm that generates contours for a two-dimensional scalar field (rectangular array of individual numerical values). A similar method can be used to contour 2D triangle meshes.

The contours can be of two kinds:

- Isolines – lines following a single data level, or isovalue.
- Isobands – filled areas between isolines.

Typical applications include the contour lines on topographic maps or the generation of isobars for weather maps.

Marching squares takes a similar approach to the 3D marching cubes algorithm:

- Process each cell in the grid independently.
- Calculate a cell index using comparisons of the contour level(s) with the data values at the cell corners.
- Use a pre-built lookup table, keyed on the cell index, to describe the output geometry for the cell.
- Apply linear interpolation along the boundaries of the cell to calculate the exact contour position. [30]

7. Conclusion

In this chapter, we presented a medical background of blood cells, and the benefits of a Complete Blood Count CBC. We conclude that there are many diseases related to blood cells, and we must aid in detecting some of them before it is too late.

We have also presented artificial intelligence, machine learning and deep learning. Also, the latest CNN architectures where we did a deeper dive into U-Net and SegNet. Finally, we defined the different image processing methods which permitted us to count blood cells.

CHAPTER II : STATE OF THE ART

1. Introduction

Blood cell segmentation is the extraction of different blood cells from microscopic images. Blood cell counting is the process of counting the detected blood cells after segmentation. Many researchers have implemented methods for segmentation and counting of blood cells using different approaches. In this section we first explore the used databases in their research and then present the methods according to the segmentation approach.

2. DataSet Collections

In this section we are comparing between the databases and their annotations that are available to pick the best database that will fit our problem. We can see that there is a lot of datasets which are created to attack a specific sets of problems, we can see below all available datasets:

2.1 ALL-IDB

ALL-IDB (Acute Lymphoblastic Leukemia Image Database for Image Processing) [24] is a public and free dataset, specifically designed for the evaluation and the comparison of algorithms for segmentation and image classification. The database focus on Acute Lymphoblastic Leukemia (ALL), Acute is a type of blood cancer that starts in white blood cells in bone marrow, the soft inner part of bones. It develops from immature lymphocytes, a kind of white blood cell that's key to immune system.[51].

Each image in the dataset, Contains classification/position of ALL lymphoblasts is provided by expert oncologists. A lymphoblast is a modified naive lymphocyte with altered cell morphology. It occurs when the lymphocyte is activated by an antigen.

The images of the dataset has been captured with an optical laboratory microscope coupled with a Canon PowerShot G5 camera. All images are in JPG format with 24 bit color depth, resolution 2592 x 1944. the ALL-IDB divides on two Datasets ALL-IDB1 and ALL-IDB2.

Dataset ALL-IDB1

The ALL-IDB1 can be used for segmentation or classification with image processing methods or Artificial intelligence models. The dataset is composed of 108 images collected during September, 2005. It contains about 39000 blood elements, where the lymphocytes has been labeled by expert oncologists.

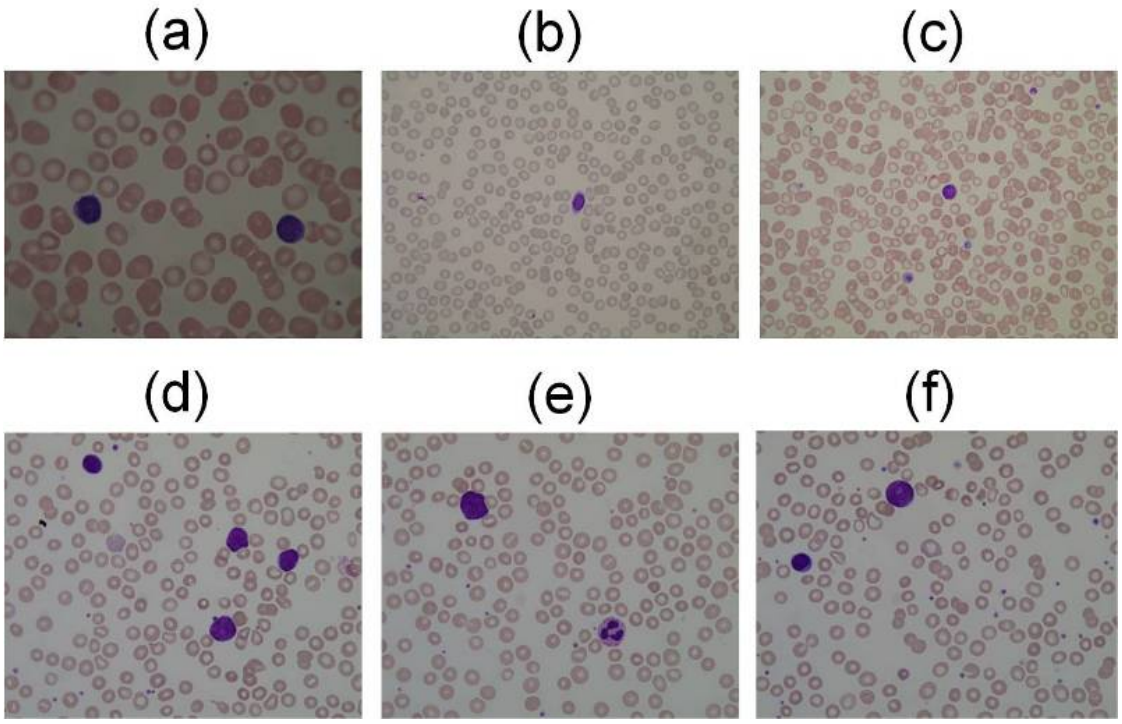
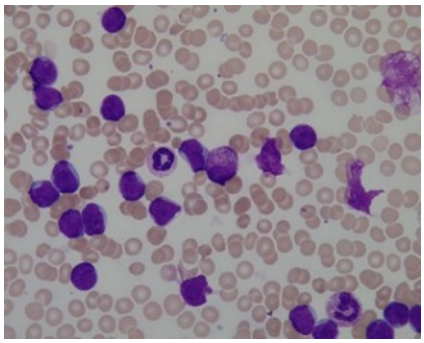


Figure 16: Examples of the images contained in ALL-IDB1: healthy cells from non-ALL patients (a,b,c), probable lymphoblasts from ALL patients (d,e,f).

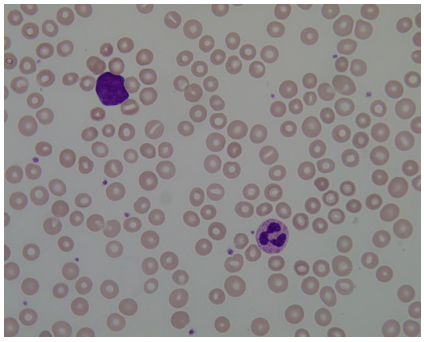
annotation: input image ”Im006_1.jpg” (a) and the related classification file ”Im006_1.xyc” reporting the coordinates of the centroids of probable ALL lymphoblasts (b).



(a) Im006_1.jpg

446	62
164	279
168	377
442	415
248	713
164	771
362	877
...	...

(b) Im006_1.xyc



(c) Im033_1.jpg

440	356

(d) Im033_1.xyc

Figure 17: Sample data from the ALL-IDB1 Database

The ALL-IDB1 image files are named with the notation ImXXX_Y.jpg where XXX is a 3-digit integer counter and Y is a boolean digit equal to 0 if no blast cells are present, and equal to 1 if at least one blast cell is present in the image. All images labeled with Y=0 are from for healthy individuals, and all images labeled with Y=1 are from ALL patients. Each image file ImXXX_Y.jpg (figure 17a, 17c) is associated with a text file ImXXX_Y.xyc (figure 17b, 17d) reporting the coordinates of the centroids of the blast cells, if any.

If we plot the coordinates in the Img006_1.xyc (figure 17a) file on the image Im006_1.jpg (figure 17b) we get the figure 18

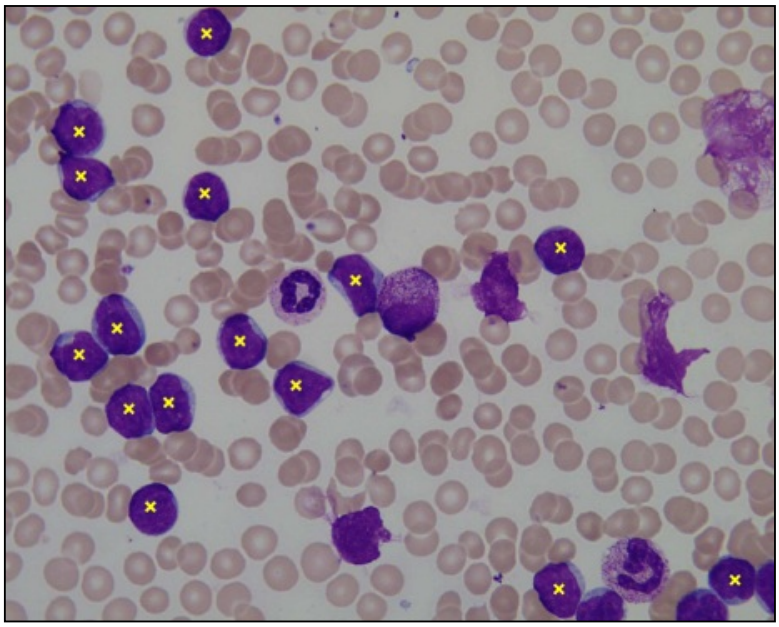


Figure 18: coordinates from Img006_1.xyc plotted on Img006_1.jpg

Dataset ALL-IDB2

This dataset has been created for testing the performances of classification systems. where the dataset has no segmentation information it contains only one information which is the presence of ALL lymphoblasts, the dataset is a collection of cropped area of interest of normal and blast cells that belongs to the ALL-IDB1 dataset as we can see in figure 19

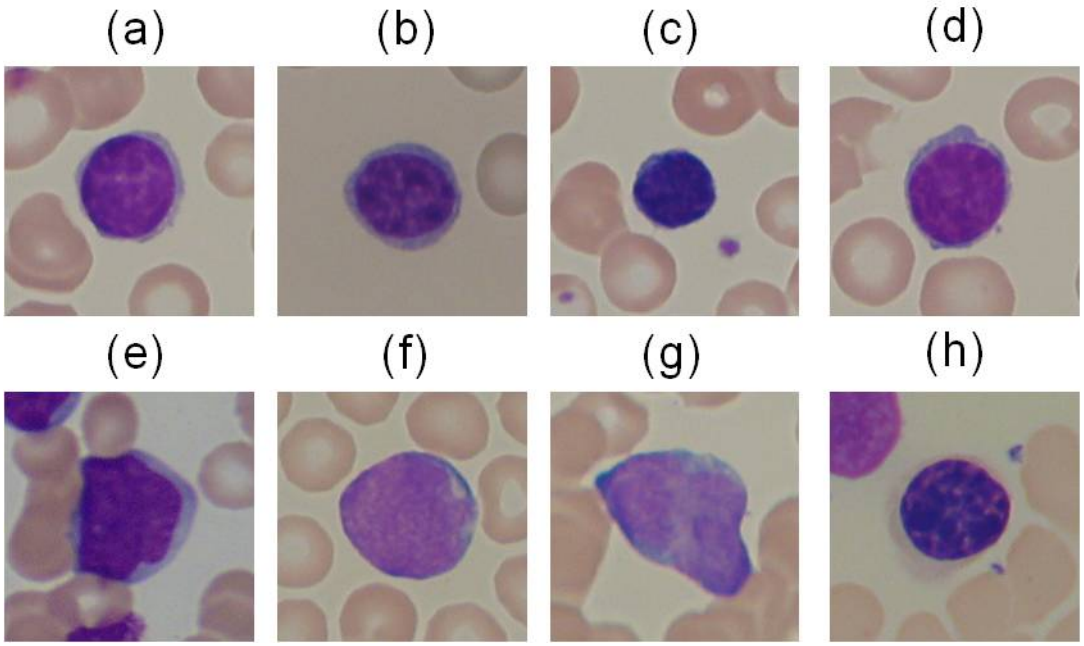


Figure 19: Examples of the images contained in ALL-IDB2: healthy cells from non-ALL patients (a-d), probable lymphoblasts from ALL patients (e-h).

The annotation of ALL-IDB2 is similar to the ALL-IDB1 but with no centroid coordinates. The ALL-IDB2 image files are named with the notation ImXXX_Y.jpg where XXX is a progressive 3-digit integer and Y is a Boolean digit equal to 0 if the cell placed in the center of the image is not a blast cell, and equal to 1 if the cell placed in the center of the image is a blast cell. all images labeled with Y=0 are from for healthy individuals, and all images labeled with Y=1 are from ALL patients.

Here is a comparison of ALL-IDB1 and ALL-IDB2:

Dataset	Algorithm type and description	Pseudocode	Verification
ALL-IDB1	Image classification The algorithm estimates if the input image is coming from a ALL patient.	CLASS = classifier ("ImXXX_Y.jpg");	The image classification is correct if and only if the output CLASS is equal to Y Alternative test: the image classification is correct if and only if the output CLASS is equal to boolean Z = M>0, where M is the number of rows in the ImXXX_Y.xyc file.
	ALL Blast cell counter The algorithm estimates the number of blast cells in the input image (if output >0, –>ALL patient).	N = blastCounter ("ImXXX_Y.jpg");	The output N is correct if equal to the M rows in the ImXXX_Y.xyc file
	ALL Blast cell identifier The algorithm estimates the location of the centroids of the blast cells in the input image (if the output cardinality is >0 –>ALL patient).	COORDINATES = blastIdentifier ("ImXXX_Y.jpg");	Error = accuracy(COORDINATES , "ImXXX_Y.xyc"); e.g., function accuracy returns the number of correct matches (for example, within a 10 pixel radius) in the COORDINATES list.
ALL-IDB2	Image classification The algorithm estimates if the input image is coming from a ALL patient.	CLASS = classifier ("ImXXX_Y.jpg");	The image classification is correct if and only if the output CLASS is equal to Y

Table 3: Comparison of ALL-IDB variants.

ALL-IDB1 UPDATED

The ALL-IDB1 Database was updated by experts where they added mask annotation for red, white and platelets for all the 108 images. The database was obtained from the shahzad paper [44].

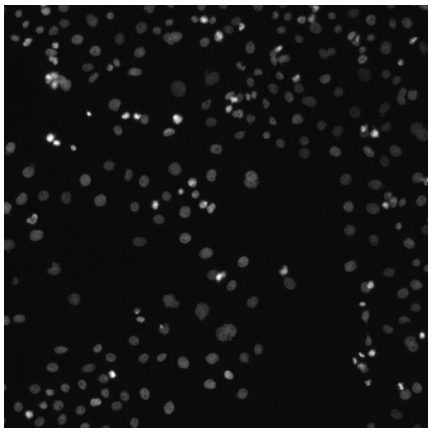
2.2 Broad Bio-image Benchmark Collection

The Broad Bio-image Benchmark Collection (BBBC) is a collection of freely downloadable microscopy image sets, cited in 450+ studies. In addition to the images themselves, each set includes a description of the biological application and some type of "ground truth" (expected results) [27]. The BBBC is organized by the Broad Institute's Imaging Platform.

The dataset contains 54 image collections of various cell types, each collection has at least one of 6 ground truth types such as cell count, foreground / background, outlines of objects, biological labels, location and bounding boxes. in the sections below we describe each ground truth:

Cell counts

In this case, the ground truth consists of the number of cells (or other objects) in each image, as counted by one or more humans. for example in BBBC1(Human HT29 colon-cancer cells) we have an image in figure 20a with the labels file in figure 20b:



(a) AS_09125_050118150001_A03f05d0.jpg

Image	manual count #1	manual count #2
AS_09125_050118150001_A03f00d0.tif	350	362
AS_09125_050118150001_A03f01d0.tif	336	342
AS_09125_050118150001_A03f02d0.tif	396	447
AS_09125_050118150001_A03f03d0.tif	320	341
AS_09125_050118150001_A03f04d0.tif	398	533
AS_09125_050118150001_A03f05d0.tif	241	257

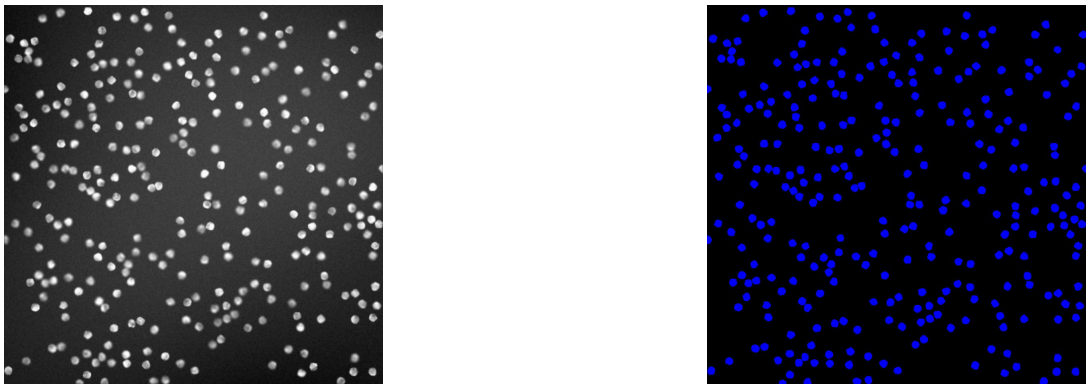
(b) BBBC001_v1_counts.txt

Figure 20: Sample data from the BBBC1 Database with counts Ground truth

in the figure 20b we can see the two counts performed by two experts, for the image 20a the first expert have found 241 cells but the second one found 257 cells.

Foreground and background

In this case, a human produces a binary (black and white but in some cases thy use other colors) image the same size as the original image. Pixels that belong to the foreground (i.e., the cells or other objects) are white, and pixels that belong to the background are black. in the example below (figure 21a and 21b) we can see the image and the corresponding mask where the cells colored with blue and the background with black.



(a) 2Gray1.tif (image) (b) 1.tif (mask)

Figure 21: Sample data from the BBBC4 Database with mask as ground truth

Outlines of individual objects

In this case, a human outlines each cell in the image in order to indicate which pixels belong to which cell. The ground truth is provided as binary images, with black outlines on a white background.

(a) 48hr-001-DIC.jpg (image) (b) 48hr-001-DIC.jpg (edge mask)

Figure 22: Sample data from the BBBC9 Database with edge mask (outlines) as ground truth

Biological labels

In these cases, the experiments have been prepared with control samples for which we know the expected biological result. The types of controls that are available dictate the type of statistic that can be calculated.

Location

In this case, the ground truth consists of the X, Y, and optionally Z location of objects (typically their centroids similar to ALL-IDB1 Annotation 18) .

Bounding Boxes

Bounding boxes are rectangles completely enclosing an object.

2.3 WBC Image Dataset : Fast and Robust Segmentation of White Blood Cell Images by Self-supervised Learning

This is two datasets of white blood cell (WBC) images used for "Fast and Robust Segmentation of White Blood Cell Images by Self-supervised Learning", which can be used to evaluate cell image segmentation methods [53]. This collection contains two datasets different from each other in terms of the image color, cell shape, background, etc. The ground truth segmentation results are manually sketched by domain experts, where the nuclei, cytoplasms and background including red blood cells are marked in white, gray and black respectively.

Dataset 1

Was obtained from Jiangxi Tecom Science Corporation [48], China. It contains three hundred 120×120 images of WBCs and their color depth is 24 bits. The images were taken by a Motic Moticam Pro 252A optical microscope camera with a N800-D motorized auto-focus microscope.

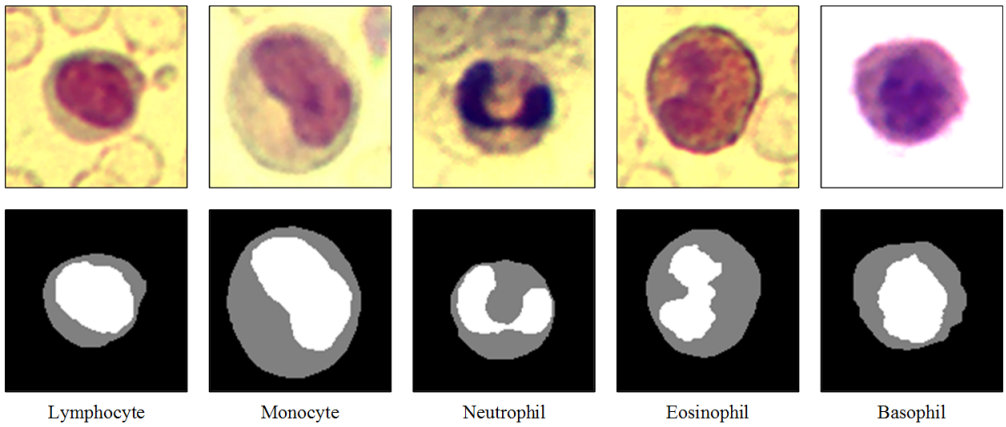


Figure 23: Sample data from WBC_Segmentaion Dataset 1

Dataset 2

Consists of one hundred 300×300 color images, which were collected from the Cella-Vision blog [7]. The cell images are generally purple and may contain many red blood cells around

the white blood cells.

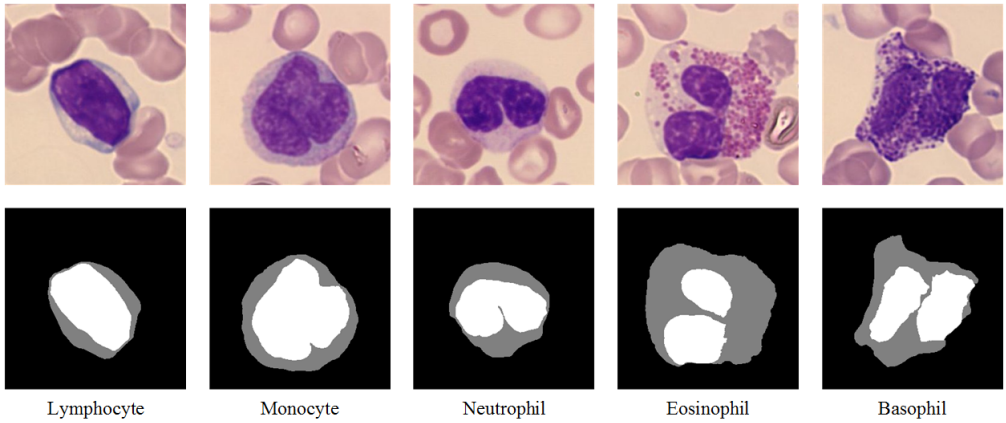


Figure 24: Sample data from WBC_Segmentaion Dataset 2

2.4 Annotation

The class labels of each image in Dataset 1 and Dataset 2 are stored in csv files. The labels (1- 5) represent neutrophil, lymphocyte, monocyte, eosinophil and basophil, respectively.

2.5 A large dataset of white blood cells containing cell locations and types, along with segmented nuclei and cytoplasm

The database is provided by RabinData and [23] which divides on two separate datasets:

Raabin-WBC Data

Contains 4 sub-Datasets:

- **Double-labeled cropped cells** : Double-labeled cropped cells are also provided containing only five main classes including mature neutrophils, lymphocytes (small and large), eosinophils, monocytes, and basophils.
- **Nucleus_cytoplasm_Ground truths** : in this sub-Dataset they prepared the ground truths of the cytoplasm and the nucleus for a proper number of cropped white blood cells. For this purpose, 1145 cropped images including 242 lymphocytes, 242 monocytes, 242 neutrophils, 201 eosinophils, and 218 basophils were randomly selected, and their ground truths were extracted by an expert.

- **Microscopic images were taken by the Olympus CX18 microscope and the Samsung Galaxy S5 camera and the 4th database with contain images taken by Zeiss microscope and the LG G3 camera** : in these two sub-Datasets. Corresponding to each microscopic image, a dictionary (json format) file containing the following information about that image was provided:
  - Information about the blood elements in the image including their coordinates and labels.
  - Information about the blood smears including staining method and the type of the disease.
  - Information about the microscope includes the type of microscope and its magnification size.
  - The type of camera used.

Raabin-Leukemia Data

Contains 4 sub-Datasets:

- **Acute Lymphoblastic Leukemia**
- **Acute Myeloblastic Leukemia**
- **Chronic Lymphocytic Leukemia**
- **Chronic Myelogenous Leukemia**

In each of these sub-Dataset, All samples were taken from patients who had referred to our collaborator medical laboratory (Takht-e Tavous Laboratory in Tehran, Iran). It should be notices Zeiss microscope and LG J3 smartphone camera had been used for imaging.

2.6 BCCD

BCCD (Blood Cell Count and Detection) Dataset is a small-scale dataset for blood cells detection. The dataset contains a total of 364 (640x480) jpeg images with their annotations. The original data and annotations are from cosmicad and akshaylamba.
In this project, the Faster R-CNN algorithm from keras-frcnn for Object Detection is used.

From this dataset, nicolaschen1 developed two Python scripts to make preparation data (CSV file and images) for recognition of abnormalities in blood cells on medical images.

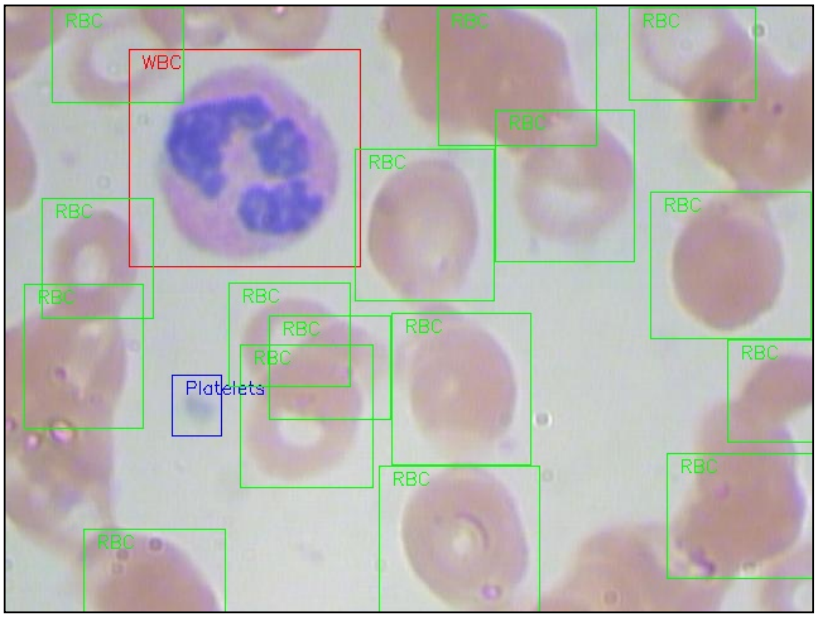


Figure 25: Sample data from BBCD

In this database they use bounding boxes to locate the cells and each bounding box has the type of cell RBC or WBC and Platelets. they are using the VOC format as a database architecture.

2.7 Dataset collections resume

Dataset Name	Dataset Size	Type of cells (RBC, WBC, Platelets)			Type of annotation	Description
ALL-IDB1 [43]	108	108 masks / 13 edges	108 masks	108 masks	Location (Centroid of cells)	Acute Lymphoblastic Leukemia Image Database for Image Processing
BBBC [27]	Lots of subdatasets	X	X	X	All types of annotations	Broad Bioimage Benchmark Collection
WBC Image Dataset	400		X		Foreground and Background (mask)	WBC Image Dataset : Fast and Robust Segmentation of White Blood Cell Images by Self-supervized Learning
BCCD [45]	364		x		Bounding boxes	Blood Cell Count and Detection

Table 4: Table that resumes datasets mentioned above

3. Computer-assisted blood cell segmentation and counting review

Bhavnani et al. [4] have developed a method for segmenting and counting RBC (red blood cells), WBC (White blood cells) and platelets which is also called complete blood count (CBC), by using Otsu's thresholding and morphological operations as a segmentation method, and for counting the are preforming a comparison between two methods: the watershed algorithm and Circular Hough Transform. The model takes an RGB image as an input apply some processing steps then uses Otsu's thresholding to extract RBC and WBC separately with different threshold values then apply the two algorithm to compare the results, the model has no image size constraint because it's based only on image processing techniques and needs a small database to select the threshold values for RBC and WBC in this article they used 20 images. In the Experiment phase they used ALLIDB Database which contains 108 images with 1712x1368 and 2592x1944 resolution.The CHT method is the best in terms of accuracy with 92.67% but it has some weaknesses with overlapping cells and morphological abnormalities. In the other side the watershed method which is a little bit adapted with overlapping and touching cells had an accuracy of 91.07%.

Guiliang, FENG et al. [15] have a developed an algorithm that segments and counts cell images based on image definition, a Discrete Cosine Transform (DCT) is applied, which is proposed by N. Ahmed and Rao in 1974 [1]. Instead of the traditional watershed approach, the DCT method showed better results in comparison.

However, there is a drawback to this approach, because this algorithm depends on image definition it relies on well focused images, consequently, when the images are out of focus the segmentation and counting is not reliable. But despite that drawback, it achieved a relatively high accuracy of over 90% which is better that the watershed method.

K. Sudha and P. Geetha [47] have developed a two stage framework which will segment the leukocytes (a type of WBC) with an edge strength-based Grabcut method as a first stage, in the second stage will count the cells using the novel gradient circular hough transform (GCHT) method. the model takes and RGB image convert it to HSV color space to extract the S component where the WBCs are more clear then applies the edge strength-based location detection the results are fed to fine segmentation using Grabcut Algorithm which will output

the edge segmentation mask, for the counting the mask will be fed to the proposed GCHT Algorithm. In the experiments phase they used ALL-IDB [24] and Cellavision [53] datasets, after resizing the images to 256x256.
After the experiments the proposed method had reached an average segmentation accuracy of 99.32% and a counting accuracy of 97.3%.
The new GCHT method can segment touched cells and even overlapped cells.

Kimbahune et al. [21] have developed a method for segmenting and counting red blood cells (RBC) and white blood cells (WBC). segmentation is done using Pulse-Coupled Neural Network (PCNN) and square tracing algorithm for contour tracing after de-noising it with PCNN combined with median filter, the counting is performed by scanning the image and using edge detection methods as square tracing algorithm. this method gave good results compared to state of art methods.

Carlos X. Hernández et al. [16] have used a convolutional neural network (CNN) using a feature pyramid network (FPN) combined with a VGG style neural network for segmenting and counting of cells in a given microscopy image. The dataset they used is BBBC005 [27] from Broad Institute's Bio-image Benchmark Collection, which consists of 9600 images and each image is 696x520 pixels but they were scaled down to 256x192 for the purposes of their experiment.

Out of the total 9600 images only 600 of the images which have a corresponding mask were used for the FPN training. And 100 of those were used for fast prototyping and a standard of 80-20 train/test split for the final models. On the other hand, the full 9600 images were used for the VGG network. This approach achieved a relatively good accuracy of 81.75% but with some failure cases such as:

- High cell overlap
- Irregular cell shapes
- bad focal planes.

Tran, Thanh and Minh et al. [49] have developed a method for segmenting and counting RBC and WBCs by using the SegNet model initialised with weights from a pre-trained VGG-16 model, for the counting they first apply Distance transform with 4 different distance metrics, then they apply binary dilation. At the End, they apply the

connected component labeling algorithm to count the number of separated cells in images mask. for the training they used 42 images from ALL-IDB1 [24] after they cropped them to decrease the computation time and memory usage and reduce the number of RBC compared to WBC the result images have a resolution of 360*480*3 (RGB), they used 29 images for training and 13 for testing ,the model had a segmentation accuracy of 89% and counting accuracy of 93.3% on RBC and accuracy of 100% on WBC with the testing database which has the cropped images of RBC and WBC.

On the first database with cropped images they had only few WBC but in this second database they have more WBC , the database 2 contains 108 only WBC images with the same size of database 1 360*480*3, they augmented the training dataset from 76 to 380 and used 32 images for testing, this second model focus only on the WBC which will increase the segmentation accuracy to 98.5%, and have a counting accuracy of 97.29%. The counting accuracy have decreased because of the clumped cells which is the weakness of this model.

Yan Kong et al. [22] have developed a two-stage framework using parallel modified U-Nets together with seed guided water-mesh algorithm for automatic segmentation and yeast cells counting which is used to observe the living conditions and survival of yeast cells under experimental conditions.

The cell images used in this study were captured by Olympus IX83 (Olympus Life Sciences, Tokyo, Japan) inverted microscope. They manually selected 20 raw DIC (differential interference contrast) images which contained a number of yeast cells and the annotations were done manually by laboratory technicians, they then obtain 40 images, 20 masked annotation images and the other 20 is center annotation images of yeast cells.

After splitting images into tiles of size 224x224 with a step stride of 65 and 33 pixels for the horizontal and vertical direction, respectively. They got 4360 raw image tiles and the corresponding center annotation and masked annotation images, from that set 3310 tiles were randomly selected as the training data set and rest was a validation set.

The raw test DIC images used in this study were sized approximately 1002x1998 pixels, but they were resized into 1092x2084 pixels so that each DIC image could be split into a grid of 8x16 image blocks. The image blocks were then fed into modified U-Net.

This method achieved a precision of over 99.74% and an average recall rate of 99.35%.

however, there is a limitation using this approach, which is the detection of small objects.

Shahzad, Muhammad et al. [44] have developed a custom convolutional encoder-decoder framework along with VGG-16 as the pixel-level feature extraction model to address the problem of whole-slide cell segmentation using the semantic segmentation approach. Their proposed framework works as follows: First, all the original images along with manually generated ground truth masks of each blood cell type are passed through the pre-processing stage. In the pre-processing stage, pixel-level labeling, RGB to gray-scale conversion of masked image and pixel fusing, and unity mask generation are performed. After that, VGG16 is loaded into the system, which acts as a pre-trained pixel-level feature extraction model. Finally, the training process is initiated on the proposed model.

They used ALL-IDB1 as their baseline dataset which consists of 108 whole-slide blood cell images, 59 (2592x1944) images were from healthy individuals and 49 (1712x1368) images from acute lymphoblastic leukemia (ALL) patients.

This approach achieved a class-wise accuracy of 97.45%, 93.34%, and 85.11% for RBCs, WBCs, and platelets, respectively, while global and mean accuracy remain 97.18% and 91.96%, respectively.

Overton, Toyah and Tucker, Allan [38] have developed a method which segments and counts IDP (Internally Displaced people) and erythrocytes (red blood cells) using DO-U-Net (Dual Output U-Net) which outputs a segmentation mask and an edge mask then they subtract them to get rid of the overlapping and the touching problem, the model trains on extremely small datasets (10 images) and gives a high segmentation accuracy, They selected 10 images of 108 from ALL-IDB dataset for training the model, the model takes images with a resolution of 188x188 and outputs a segmentation mask and edge mask of lower resolution 100x100, the experiments results have given an accuracy of 98.31% on a 5 randomly selected images from ALL-IDB, for the IDP they had 98.69% for fixed resolution images and 94.66% for scale-invariant satellite images.

Li, Dongming et al. [25] have developed a method for segmenting blood cells by combining neural ordinary differential equations (NODEs) with U-Net networks to improve the accuracy of image segmentation. In order to study the effect of ODE-solve on the speed

and accuracy of the network, the ODE-block module was added to the nine convolutional layers in the U-Net network. Firstly, blood cell images are pre-processed to enhance the contrast between the regions to be segmented; secondly, the same dataset was used for the training set and testing set to test segmentation results. Then they select the location where the ordinary differential equation block (ODE-block) module is added, select the appropriate error tolerance, and balance the calculation time and the segmentation accuracy, in order to exert the best performance.

Finally, the error tolerance of the ODE-block is adjusted to increase the network depth, and the training NODEs-UNet network model is used for cell image segmentation.

The experiment dataset for this model was provided by the Center for Medical Image and Signal Processing (MISP) and the Department of Pathology, Isfahan University of Medical Sciences [42]. MISP.rar contains 148 clear blood cell smear images with a size of 775x519 pixels. They picked up appropriate areas for convenient network training, then cropped 100 blood cell images with a size of 256x256 pixels by selecting a suitable area. To ensure the accuracy of the training model, they retained 20 images as the testing set and used the remaining 80 images to increase the dataset to 800 by data augmentation. Besides, they used a ratio of 3 : 1 as the training set and the validation set.

Using this approach to segment blood cell images in the testing set, it can achieve 95.3% pixel accuracy and 90.61% mean intersection over union. By comparing the U-Net and ResNet networks, the pixel accuracy of this network model is increased by 0.88% and 0.46%, respectively, and the mean intersection over union is increased by 2.18% and 1.13%, respectively.

4. Comparative study

Reference	Segmentation Approach	Counting Approach	Image Size	Segmentation Accuracy	Counting Accuracy	Dataset Size	Dataset Name	Targeted Cells
Kimbahune et al. (2011) [21]	PCNN	Square tracing method	N/A	N/A	N/A	N/A	N/A	Red/White/Platelets
Guiliang, FENG et al. (2016) [15]	Discrete Cosine Transform (DCT)		N/A	90%		N/A	N/A	Red/White/Platelets
Bhavnani et al. (2016) [4]	OTsu's thresholding and morphological operations	1-Circular Hough Transform 2-Watershed Algorithm	2594x1944 1712x1368	1- 92.67% 2- 91.07%		20 / 108	ALLIDB1 [43]	Red/White
Carlos et al. (2018) [16]	FPN combined with VGG style neural network		256x196	95%		80 / 20	BBBC005	counts the total number
Tran, Thanh and Minh et al. (2019) [49]	SegNet with weights from a pre-trained VGG-16	Distance Transform and connected component labeling algorithm	360x480x3 (RGB)	98.5%	97.29%	380 / 32	ALL-IDB1 [43]	Red/White
K. Sudha and P. Geetha (2020) [47]	Edge strength-based Grabcut	Gradient Circular Hough Transform	256x256	99.32%	97.3%	1210	ALL-IDB1 [43]	White
Yan Kong et al. (2020) [22]	Two parallel modified U-Nets	Seed Guided Water-Mesh Algorithm	224x224	96%		3310 / 1050	Self Annotated	counts the total number
Shahzad et al. (2020) [44]	Custom encoder-decoder framework with VGG-16	N/A	2594x1944 1712x1368	97.45% 93.34%	N/A	108	ALL-IDB1 [43]	Red/White/Platelets
Overton, Toyah and Tucker, Allan (2020) [38]	DO-U-Net	Marching Squares Algorithm	188x188	98.31%		10 / 5	ALL-IDB1 [43]	Red
Li, Dongming et al. (2021) [25]	Neural Ordinary Differential Equations (NODEs) with U-Net networks	N/A	256x256	95.3%	N/A	800	MISP[42]	White

Table 5: Table that represents a comparative study of previous methods

We can see in the table 5 each method with the approach they used, the accuracy that they achieved and the type of cells that they are targeting.

5. Conclusion

First in this chapter, we explored all the available datasets and their annotations. Later, we saw the diversity of state of the art methods where most approaches have used deep learning as a segmentation method paired with multiple image processing methods for counting blood cells.

Chapter III: Contribution

1. Introduction

As part of our research, we have treated the case of segmenting and counting Red, White blood cells and platelets which also known as CBC (Complete Blood Count), we are using the ALL-IDB[43] Dataset to train and evaluate our models. In our case study, and from multiple articles, we can see that U-Net and Segnet models are dominating the field of cell segmentation and Medical Computer vision in general. we've chosen the article of Overton [38] because of the performance and optimisation of their segmentation model and we applied the same idea on the SegNet model, and for the counting methods we took the 3 of the most used methods to compare between them.

In this chapter, we test-out both U-Net and Segnet models, and analyse the results by comparing results of the two architectures. We will also explore different machine learning algorithms for both pre-processing and post-processing.

2. Proposed approach

From all of the intel we have gathered, and previously read articles, all cell segmentation tasks (blood cell segmentation in particular) are mostly using U-Net and SegNet architectures for segmenting blood cell images. We have used both the U-Net and SegNet models. In the following sections, we will briefly analyze and compare both convolutional neural network (CNN) models with their perspective results. And explain all the postprocessing methods we used for the counting of blood cells (red, white and platelets).

3. DO-UNet

3.1 Definition

The U-Net is a convolutional neural network that was developed for biomedical image segmentation at the Computer Science Department of the University of Freiburg. The network is based on the fully convolutional network and its architecture was modified and

extended to work with fewer training images and to yield more precise segmentations. In our case we are using DO-UNet from [38] which is a modified U-Net to produce dual outputs, which also known as contour aware network was first demonstrated by the DCAN architecture [8]. Based on a simple FCN, DCAN was trained to use the outer contours of the areas of interest to guide the training of the segmentation masks. This led to improved semantic and instance segmentation of the model, which in their case, looked at non-overlapping features in biomedical imaging. With the aim of counting closely co-located and overlapping cells, we are predominantly interested in the correct detection of individual objects as opposed to the exact precision of the segmentation mask itself. An examination of the hidden convolutional layers of the classical U-Net showed that the penultimate layer of the network extracts information about the edges of the cells, so the idea is to output the cell mask + edge mask then do a substraction to break the overlapping cells.

3.2 Architecture

They Started with the classical U-Net then reduced the number of convolutional layers and skip connections in the model. Simultaneously, they minimised the complexity of the model by looking at smaller input regions of the images, thus minimising the memory footprint of the model. They follow the approach of Ronneberger et. al. [38] by using unpadded convolutions throughout the network, resulting in a model with smaller output edge and mask (100 $\times$ 100 px) corresponding to a central region of a larger (188 $\times$ 188 px) input image region. DO-U-Net uses two, independently trained, output layers of identical size. Figure 26 shows the DO-U-Net architecture.

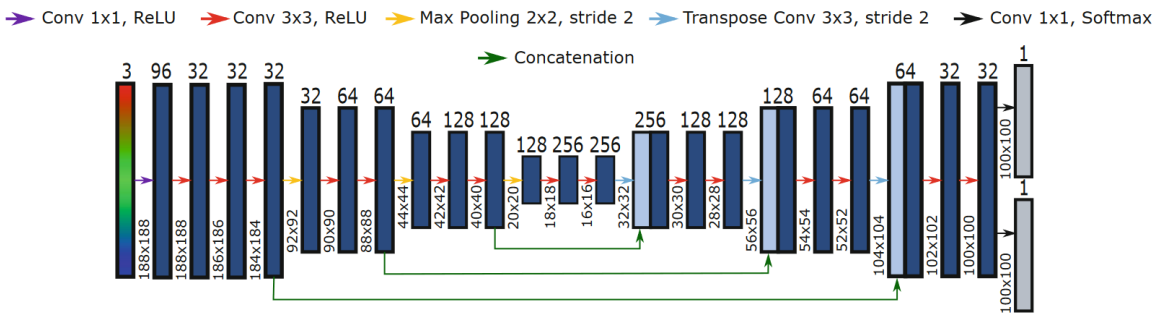


Figure 26: DO-UNet architecture

4. Segnet

4.1 Definition

The SegNet neural network, developed by Alex Kendall, Vijay Badrinarayanan, and Roberto Cipolla, all from the University of Cambridge, is a convolutional neural network used for semantic pixel wise labeling. This problem is more commonly called semantic segmentation. [3]

4.2 Architecture

SegNet has an encoder network and a corresponding decoder network, followed by a final pixelwise classification layer. This architecture is illustrated in the figure below. The model we used has the 13 encoder layers obtained from the VGG16 network, and 13 decoder layers to match the same number of encoder layers. The final decoder output is fed to a multi-class soft-max classifier to produce class probabilities for each pixel independently (pixelwise).

Each encoder in the encoder network performs convolution with a filter bank to produce a set of feature maps. These are then batch normalized. Then an element-wise rectified-linear non-linearity (ReLU) max(0, x) is applied. Following that, max-pooling with a 2x2 window and stride 2 (non-overlapping window) is performed and the resulting output is sub-sampled by a factor of 2. Max-pooling is used to achieve translation invariance over small spatial shifts in the input image.

The appropriate decoder in the decoder network upsamples its input feature map(s) using the memorized max-pooling indices from the corresponding encoder feature map(s). This step pro- duces sparse feature map(s). This SegNet decoding technique is illustrated in the below figure. These feature maps are then convolved with a trainable decoder filter bank to produce dense feature maps. A batch normalization step is then applied to each of these maps. Note that the decoder corresponding to the first encoder (closest to the input image) produces a multi-channel feature map, although its encoder input has 3 channels (RGB). This is unlike the other decoders in the network which produces feature maps with the same number of size and channels as their encoder inputs. The high dimensional feature representation at the output of the final decoder is fed to a trainable soft-max classifier. [3]

The input shape we used is (128x128x3) and the output shape is (128x128), there is no loss in resolution because segnet uses the option ‘same’ padding on each encoder layer. We also tested 3 different loss functions (tversky loss, binary crossentropy and Mean Squared Error MSE), out of which the MSE outperformed the others.

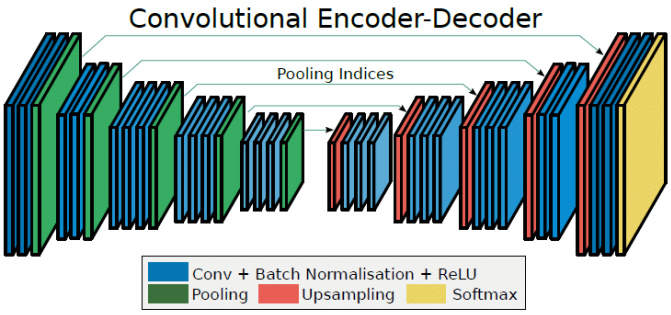


Figure 27: SegNet architecture

5. Dataset

For both models, we decided to work with the updated ALL-IDB1 dataset which has 13 RBC edges and masks, 108 WBC, Platelets Masks and 13 RBC images which has the count information, but the WBC and Platelets don’t have the count information where we had to use manual count and algorithms to find the count information for these images.

10 images with their perspective masks and edge masks were chosen for red blood cell training and 3 as a test dataset, for white blood cells 73 images with their masks, and 33 as a test dataset. For platelets, we used 71 for training and 31 as a test dataset. Only red blood cells have edge masks, because we need to get rid of overlapped cells, white blood cells and platelets dont need the edge masks, using masks only can retrieve all the necessary features, because both white blood cells and platelets rarely overlap. The images will be sliced to the input size of the according model to match the input shape of the models. The resulting train dataset will be 3916 image, mask, and edge tiles (a total of 11748 tiles). For the test dataset 1072 image, mask, and edge tiles (a total of 3216 tiles) for red blood cells. As for white blood cells, 28126 image and mask tiles were used for training (a total of 56252 tiles), and 15892 image and mask tiles were used as a test dataset for white blood cells (a total of 31784 tiles). Finally, for platelets we used 27650 image and mask tiles were used for training (a total of 55300), and 14410 image and mask tiles as a test dataset for platelets (a total of 28820).

Dataset		Train images	Test images	Train Tiles	Test Tiles	Total images	Total tiles
Red Blood Cells	**Image**	10	3	3916	1072	13	**4988**
	Mask	10	3	3916	1072	13	**4988**
	Edge	10	3	3916	1072	13	**4988**
White Blood Cells	**Image**	73	33	28126	15892	106	**44018**
	Mask	73	33	28126	15892	106	**44018**
Platelets	**Image**	71	31	27650	14410	102	**42060**
	Mask	71	31	27650	14410	102	**42060**

Table 6: Dataset used for both models

6. Dataset augmentation

We used the same dataset augmention on all cells (red, white blood cells, and platelets) in both models UNet and SegNet. The augmentation we used was custom which involves the following steps:

1. Pick a random image from the train dataset.
2. Get the x and y coordinates randomly from the chosen image.
3. Rescale the image randomly to a smaller size then scale it back to the original size to reduce quality.
4. Take a slice of the image and mask accordingly and also edge if available.
5. Skip the image if it dosn't contain out object of interest
6. Resize the image and mask to the model input.
7. Randomly rotate and flip the image chip.
8. Randomly augment the colors (luminosity and saturation).

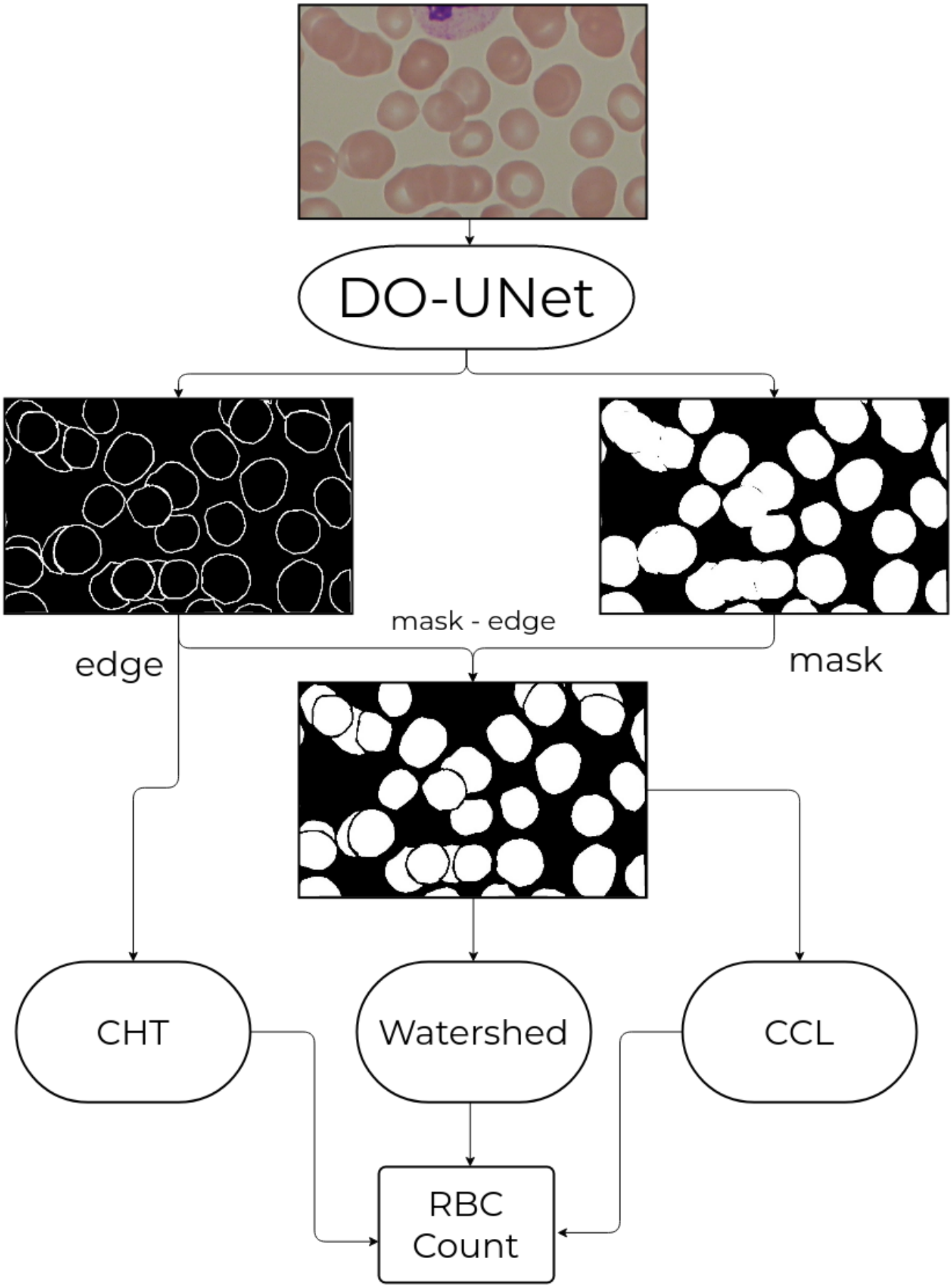


Figure 28: Schema of the segmentation and counting steps of the RBC's

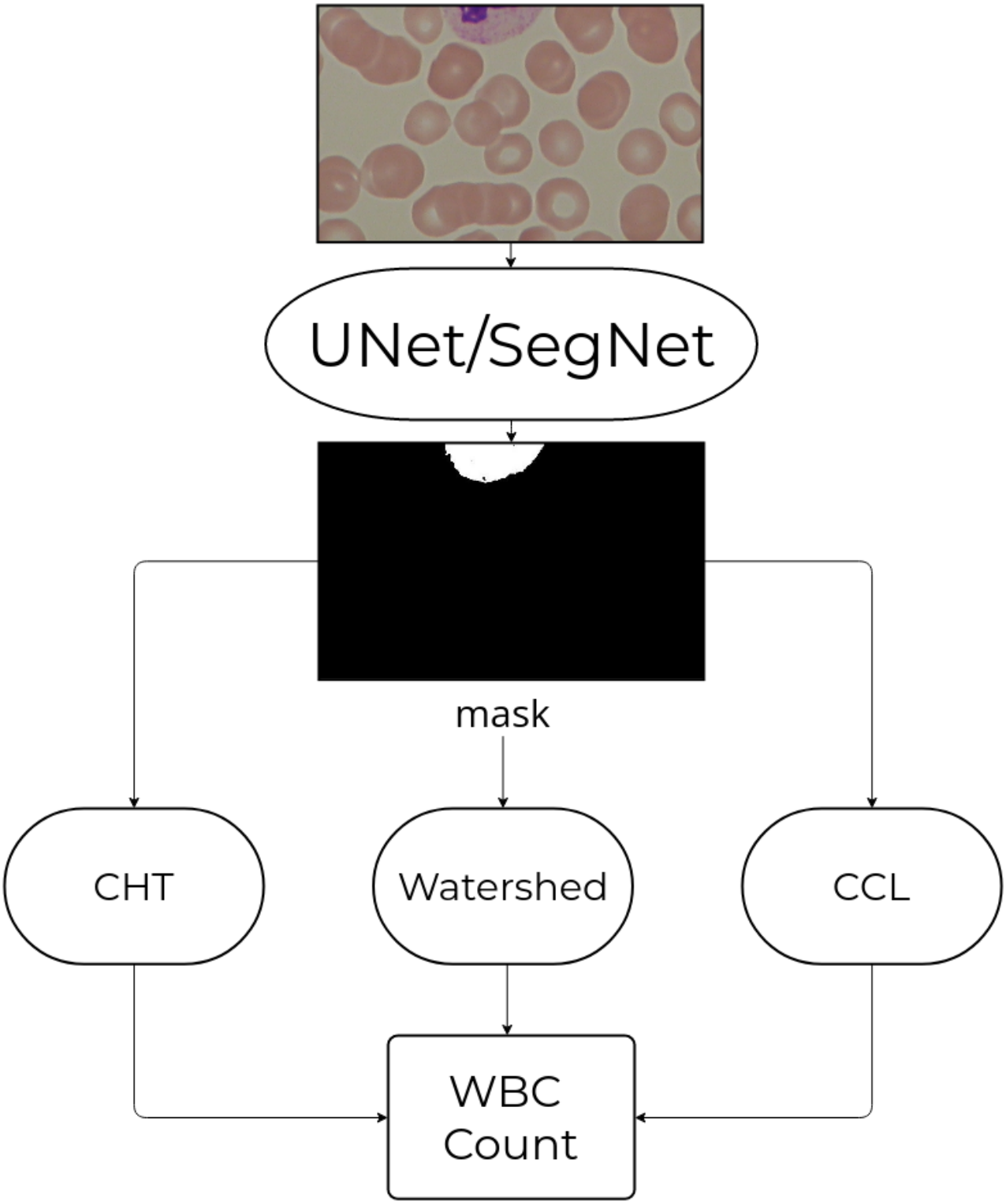


Figure 29: Schema of the segmentation and counting steps of the WBC's and Platelets

7. Counting

After having segmented blood cell images (red, white blood cells and platelets), we use multiple post-processing methods (machine learning algorithms) to get the coordinates of circles and count them. as we can see in fig 28 and 29.
We can see below are all the algorithms we used to get an relatively accurate blood cell count:

7.1 Circle Hough Transform

Circle Hough Transform (CHT) is machine learning algorithm used to extract features (circles) from imperfect images. We modified its parameters (Minimum distance, Minimum and Maximum radius...) for each type of blood cells (red, white and platelets).
Note that we do not rely on this approach to count white blood cells, because most white blood cells have different shapes. Therefore, this method is useless when it comes to white blood cells counting.

We Modified this method by adding a loss function which will help us to eliminate False Positives circles by calculating the percentage of the intersection between the circle and the cell mask. we improved the counting accuracy by more than 20% with a threshold intersection percentage of 60%.
As we can see in fig 30 the steps of the counting with the CHT method:

- we first take the mask/edge from the model.
- we apply a threshold on the mask to binarize it.
- we apply our surface filter algorithm to filter object that are not in the size range of the cell that we are counting.
- apply the circle hough transform to detect the circles in the cleaned image.
- feed the binary mask and the obtained circles from CHT to calculate loss of each circle (percentage of intersection).
- return the final number of circles that meet the threshold condition which is the circle count.

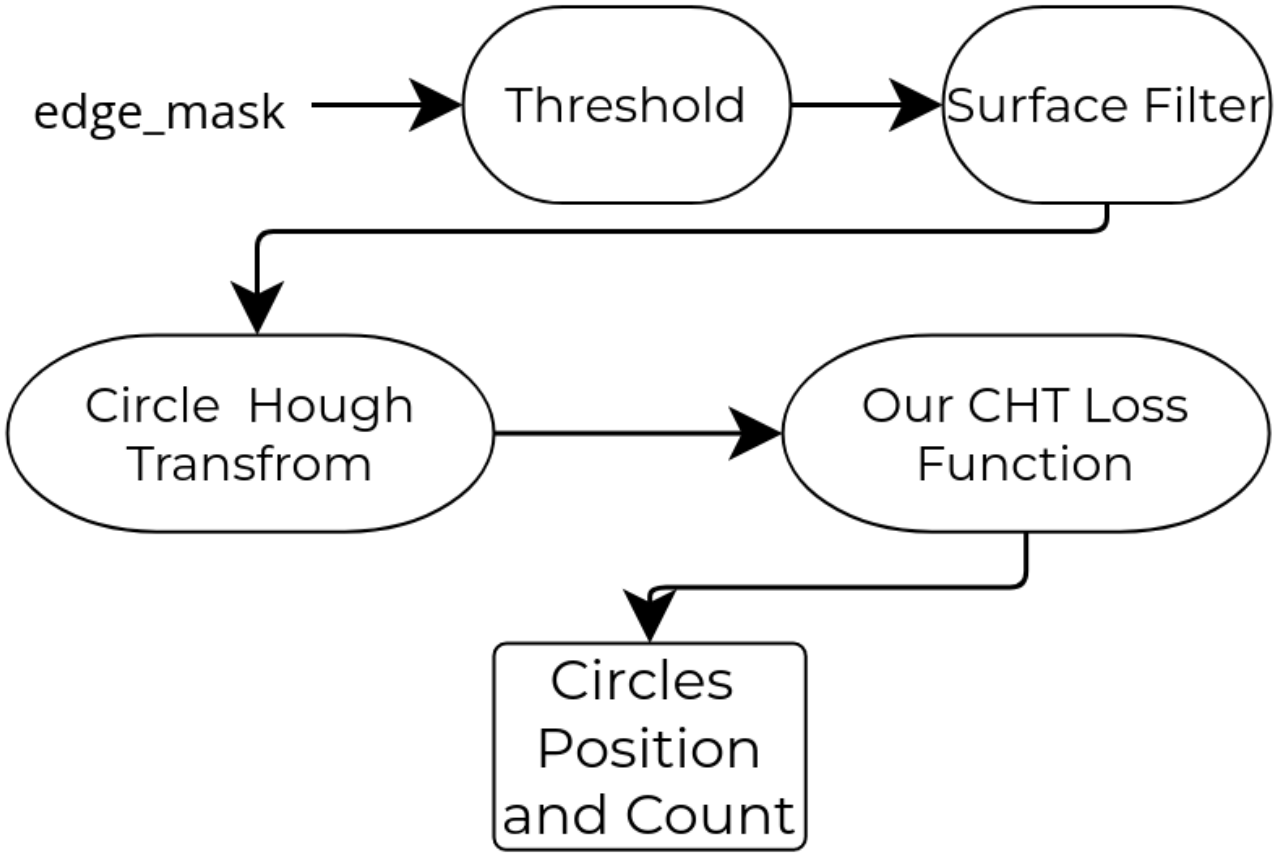
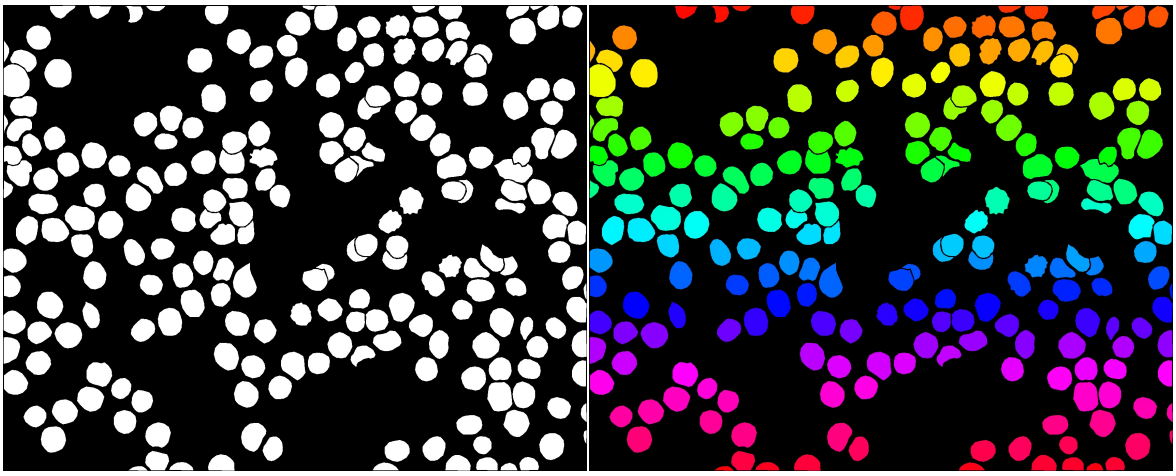


Figure 30: CHT schema applied to count blood cells

7.2 Connected Component Labeling

Connected Component Labeling (CCL) is machine learning algorithm used to detect connected regions in a binary image. Before applying the connected component labeling, we convert the images to gray-scale. Then, a binary threshold is applied to the images to get binary values. Finally, we apply the connected component labeling to get the labels and map component labels to the resulting image, and the number of labels is the cell count accordingly.



(a) input Image (b) output image

Figure 31: Example of connected component labeling

7.3 Watershed

Watershed algorithms (also called drainage divide) are used in image processing primarily for object segmentation purposes, that is, for separating different objects in an image. the main purpose of using watershed in this phase is to segment the touching and overlapping cells, the watershed takes two inputs, first i takes an image with different intensity levels in our-case the distance transform of our mask where the intensity levels represents reliefs. the second input is the water sources in our-case we extracted local maxima from the distance transform image. we can see below the steps we used to count the cells.

1. **Compute the Euclidean distance**: we compute euclidean distance from every binary pixel to the nearest zero pixel, this map will be used as our relief map in the watershed algorithm.

2. **We find peaks in the distance map**: we search for peaks in our euclidean distance map which is the local maxima in each region, which are the highest points in the map (higher intensity levels), which we will use as water sources in the watershed algorithm.

3. **Apply connected component labeling on the peak map**: we apply CCL algorithm which is also called 8-connectivity algorithm to label the peaks (label each water source).

4. **Apply the Watershed algorithm on the reversed distance map using the labeled peaks**: at the end we feed the reversed distance map and the water sources map (local maxima) to the watershed algorithm to get the segmented image.

Here is a schema presenting the previously mentioned steps:

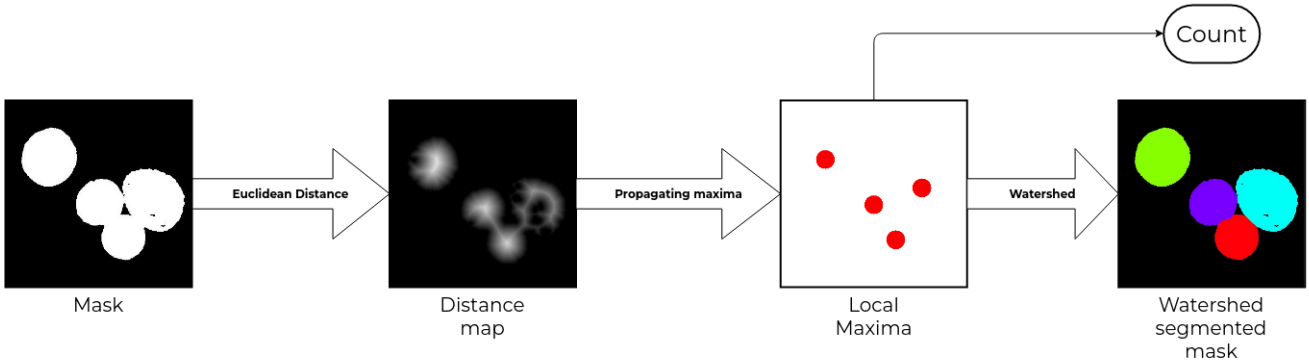


Figure 32: Watershed schema applied to count white blood cells

8. Metrics And Loss Functions

Loss functions are one of the important ingredients in deep learning-based medical image segmentation methods. In the past four years, more than 20 loss functions have been proposed for various segmentation tasks. Most of them can be used in any segmentation tasks in a plug-and-play way, we can see in (fig 33) the relations between the most used Loss Functions, we will present below the used Loss Functions in our paper.

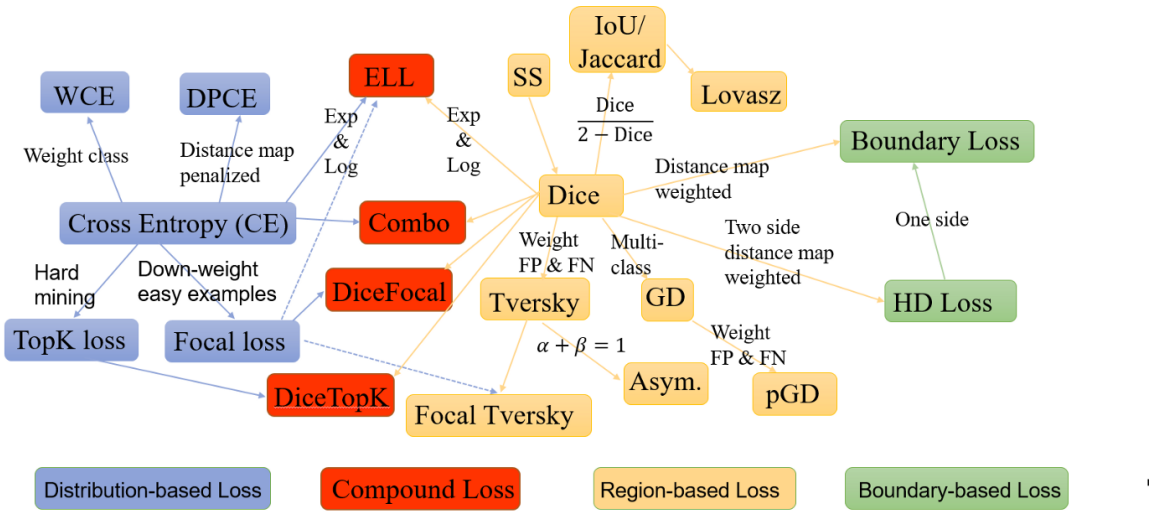


Figure 33: Loss Functions

In this section, we will discuss the loss functions and metrics that we used to train and evaluate our models.

When performing classification predictions (pixel-wise classification in our case) there's four types of outcomes that could occur.

1. **True positives** are when you predict an observation belongs to a class and it actually does belong to that class.

2. **True negatives** are when you predict an observation does not belong to a class and it actually does not belong to that class.

3. **False positives** occur when you predict an observation belongs to a class when in reality it does not.

4. **False negatives** occur when you predict an observation does not belong to a class when in fact it does.

These four outcomes are often plotted on a confusion matrix. The following confusion matrix is an example for the case of binary classification. This matrix should be generated making predictions on the test data and then identifying each prediction as one of the four possible outcomes described above.

		Actual Values	
		Yes (1)	**No (0)**
Predicted Values	**Yes (1)**	**TP**	**FP**
	No (0)	**FN**	**TN**

Table 7: Confusion Matrix

The three main metrics used to evaluate a classification model are accuracy, precision, and recall.

Accuracy is defined as the percentage of correct predictions for the test data. It can be calculated easily by dividing the number of correct predictions by the number of total predictions.

$$Accuracy = \frac{Correct\ Predictions}{All\ Predictions} \tag{5}$$

Precision is defined as the fraction of relevant examples (true positives) among all of the examples which were predicted to belong in a certain class.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \tag{6}$$

Recall is defined as the fraction of examples which were predicted to belong to a class with respect to all of the examples that truly belong in the class.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \tag{7}$$

We have a semantic segmentation problem. Therefore, we use the following metrics:

8.1 Pixel Accuracy

Pixel accuracy is perhaps the easiest to understand conceptually. It is the percent of pixels in the input image that are classified correctly.

We don't rely on this metric because it is susceptible to class-imbalance, which is when the classes are extremely imbalanced, it means that a class or some classes dominate the image, while some other classes make up only a small portion of the image. Unfortunately, class imbalance is prevalent in many real world data sets, so it can't be ignored.

To further illustrate this, if an input image was 100% black, the output prediction would be above 90% accurate, which is a totally false prediction as presented in the figure below.

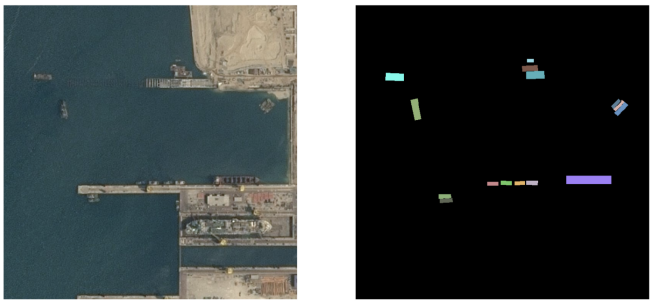


Figure 34: Example class imbalance

8.2 IOU

The Jaccard Index or Intersection Over Union, also known as the Jaccard similarity coefficient, is a statistic (metric) used for gauging the similarity and diversity of sample sets. It was developed by Grove Karl Gilbert in 1884 as his ratio of verification (v),[35]

and now is frequently referred to as the Critical Success Index in meteorology. It was later developed independently by Paul Jaccard, originally giving the French name 'Coefficient de Communauté'. [20] The Jaccard coefficient measures similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets, here is the formula:

$$J(A,B) = \frac{Area\ of\ Overlap}{Area\ of\ Union} = \frac{|A \cap B|}{|A \cup B|} \tag{8}$$

Here is an example of using IOU on a stop sign, where the green bounding box is the ground truth (the right prediction) and the red bounding box is what the model predicted.

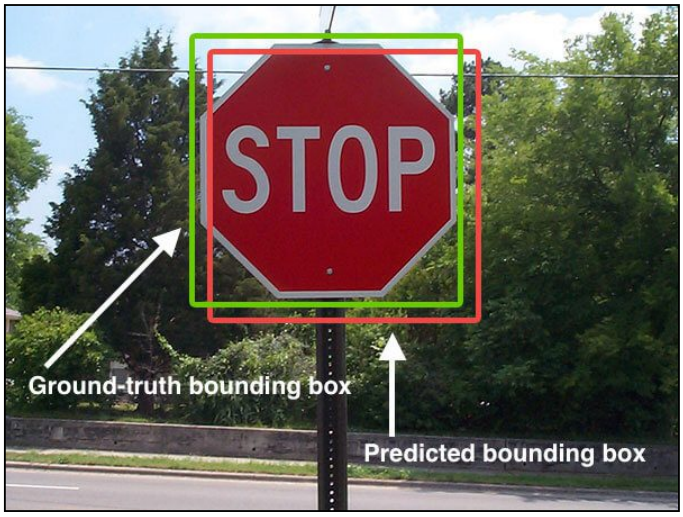


Figure 35: Example of IOU applied on a stop sign image

8.3 Dice

The Sørensen–Dice coefficient is a statistic used to measure the similarity of two samples, It was developed by the botanists (scientific study of plants) Thorvald Sørensen and Lee Raymond Dice, who published in 1948 and 1945 respectively.
Sørensen's original formula was intended to be applied to discrete data. Given two sets, X and Y, it is defined as :

$$DSC(X,Y) = \frac{2|X \cap Y|}{|X| + |Y|} \tag{9}$$

where |X| and |Y| are the cardinalities of the two sets . The Sørensen index equals twice the number of elements common to both sets divided by the sum of the number of elements in each set as we can see in fig 36. When applied to Boolean data, using the definition of true

positive (TP), false positive (FP), and false negative (FN), it can be written as :

$$DSC = \frac{2TP}{2TP + FN + FP} \tag{10}$$

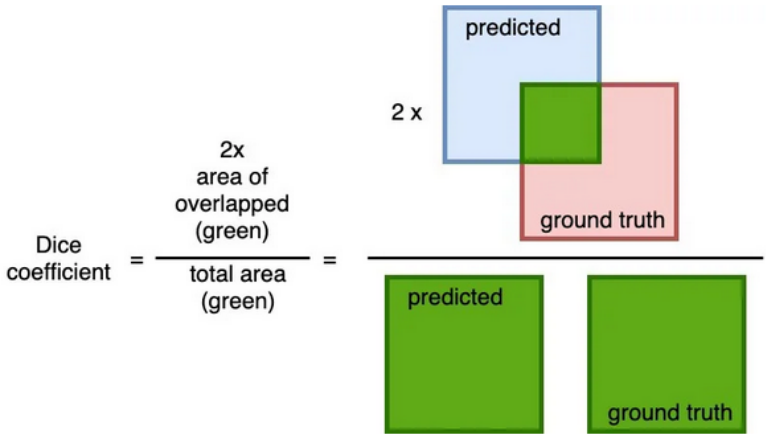


Figure 36: Example explaining Dice coefficient

This coefficient is not very different in form from the Jaccard index (IOU). In fact, both are equivalent in the sense that given a value for the Sørensen–Dice coefficient S , one can calculate the respective Jaccard index value J and vice versa, using the equations :

$$J = \frac{DSC}{(2 - DSC)} \tag{11}$$

And

$$DSC = \frac{2J}{(1 + J)} \tag{12}$$

The function ranges between zero and one, like Jaccard. the corresponding loss function:

$$DSC\_LOSS = 1 - \frac{2TP}{2TP + FN + FP} \tag{13}$$

8.4 Tversky

The Tversky index, named after Amos Tversky, is an asymmetric similarity measure on sets that compares a variant to a prototype. The Tversky index can be seen as a generalization of the Sørensen–Dice coefficient and the Dice coefficient (aka Jaccard index). For sets X and

Y the Tversky index is a number between 0 and 1 given by :

$$S(X,Y) = \frac{|X \cap Y|}{|X \cap Y| + \alpha|X \setminus Y| + \beta|Y \setminus X|} \tag{14}$$

Here, $X \setminus Y$ denotes the relative complement of Y in X. Further, $\alpha, \beta \geq 0$ are parameters of the Tversky index. Setting $\alpha = \beta = 1$ produces the Jaccard coefficient; setting $\alpha = \beta = 0.5$ produces the Sørensen–Dice coefficient.

The function ranges between zero and one, the corresponding loss function:

$$S\_LOSS(X,Y) = 1 - \frac{|X \cap Y|}{|X \cap Y| + \alpha|X \setminus Y| + \beta|Y \setminus X|} \tag{15}$$

8.5 Cross-entropy

Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1. Cross-entropy loss increases as the predicted probability diverges from the actual label. So predicting a probability of .012 when the actual observation label is 1 would be bad and result in a high loss value. A perfect model would have a log loss of 0.

The cross-entropy for a single example in a binary classification task can be stated by unrolling the sum operation as follows:

$$H(X,Y) = -(X(class0) * log(Y(class0)) + X(class1) * log(Y(class1))) \tag{16}$$

8.6 Mean Squared Error

Mean squared error (MSE) is simply defined as the average of squared differences between the predicted output and the true output. Squared error is commonly used because it is agnostic to whether the prediction was too high or too low, it just reports that the prediction was incorrect.

This is the Mean Squared Error formula:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{17}$$

We used this loss function in both our DO-U-Net and DO-SegNet models for predicting red and white blood cells.

There is however a slight problem, the Mean Squared Error has the disadvantage of heavily weighting outliers.[11] This is a result of the squaring of each term, which effectively weights large errors more heavily than small ones.

9. Conclusion

In this chapter we presented the components of our method from segmentation models and their metrics, loss functions and data augmentation. We also explained the counting methods that we are using in our solution.

CHAPTER IV : EXPERIMENTS AND RESULTS

1. Introduction

In this chapter, we will detail the different experiments and we will discuss the evaluation results obtained from the two segmentation models DO-UNet and DO-SegNet. Also, we discuss the evaluation results of three counting algorithms: Watershed, Circle Hough Transform and Connected Component Labeling for the three blood elements: RBCs and WBCs and Platelets.

2. Used Tools

2.1 TensorFlow

In our work we used TensorFlow which is an open source library developed by Google Brain Team for Artificial Intelligence, it contains multiple pre-defined models and algorithms for Deep Learning and Machine Learning, TensorFlow can be used in multiple languages as python, C++, Java and JavaScript.
TensorFlow can be used to Create, Train, Deploy Models. And when we talk about complicated models we can use Keras. Keras is a high level API for Neural Networks which helps with experiments on the models and extensibility.

Figure 37: Tensorflow

2.2 Colab

Google Colab or Colaboratory is a free Jupyter notebook environment running on Google's cloud servers for machine learning training and research. This platform allows the user to leverage back-end hardware such as GPUs and TPUs and train Machine Learning and Deep Learning models directly in the cloud. Without the need to install anything on our

computer at the anything on our computer except a browser. but it has some disadvantages where we have a limit on the GPU usage and non peresistant storage.

Figure 38: Google Colab

2.3 PaperSpace

Paperspace is a high-performance cloud computing and ML development platform for building, training and deploying machine learning models. It has a complete jupyter notebook environment which has a persistent storage and 6 hours limit for each execution.

Figure 39: Paperspace

3. do-U-Net Results

3.1 Red Blood Cells

The Red Blood Cells are the most difficult to detect because of the overlapping, where in some samples we can't notice the overlapping by eyes, in this experiment we are testing do-U-Net from [38]. In the do-U-Net we updated the data augmentation phase. and applied Transfer Learning to get better edge mask. we can see in the dataset that we have only 13 images out of 108 from ALL-IDB1 that contains edge-masks and 108 masks, so the problem here is the lack of the edge label. we ended up with the method below as the best fit to our problem:

1. Train the DO-UNet outputs on the large dataset (108 masks) which will output two identical masks.

2. Continue Training the DO-UNet with the small dataset (13 masks, 13 edges), and Freeze the Mask Output.

RBC_Model	Dataset	Epochs	Output	Loss	Mean IOU	Dice	Tversky	Accuracy
A	small Dataset (mask + edge) * (10 + 3)	800	Mask	0.3615	0.6365	0.8304	0.8232	0.8754
			Edge	0.1816	0.0663	0.3582	0.3475	0.9343
B	Phase 1: large dataset mask*108 Phase 2: small Dataset (mask + edge)*13	Phase 1: 120 Phase 2: 400	Mask	0.0713	0.7751	0.9528	0.9567	0.9716
			Edge	0.1465	0.0759	0.4127	0.4015	0.9385

Table 8: Normal trained modes compared to transfer learning model

The table 8 compares between the normal trained model A on the small dataset which contains 13 masks and edges and the model be which is trained on two phases, first with the large dataset (108 mask without edge) for 60 epochs, in the second phase we continued training with the small dataset (13 masks + edges) for 400 epochs. We can see that this method pushed the edge accuracy which is really important to get rid of the overlapping.

After training the dual-output-U-Net model we did a benchmark on the 13 images to calculate the counting accuracy of the three methods, we ended up with the table 9 below where we can see that the circle hough transform method is the best for RBC counting with 95.36 accuracy, because all the RBCs have a similar shape and size.

Image	Real count	Watershed	CCL	CHT	Watershed_acc	CCL_acc	CHT_acc
Im037_0	105	85	70	102	80.95	66.67	97.14
Im045_0	601	517	415	577	86.02	69.05	96.01
Im053_1	802	571	320	652	71.20	39.90	81.30
Im001_1	215	214	205	216	99.53	95.35	99.53
Im004_1	258	280	247	268	91.47	95.74	96.12
Im015_1	293	295	260	281	99.32	88.74	95.90
Im022_1	242	251	235	246	96.28	97.11	98.35
Im050_1	460	498	423	519	91.74	91.96	87.17
Im069_0	665	651	531	675	97.89	79.85	98.50
Im079_0	512	496	398	518	96.88	77.73	98.83
Im095_0	150	189	95	150	74.00	63.33	100.00
Im099_0	528	478	397	517	90.53	75.19	97.92
Im108_0	510	511	418	546	99.80	81.96	92.94
Total					90.43	78.66	95.36

Table 9: RBC Counting results using 3 algorithms

3.2 White Blood Cells

The White Blood Cells are also difficult to detect because of the non stable shape and in some cases they overlap, in this experiment we are comparing single output U-Net from [38] and the single output SegNet model. In the U-Net we removed the edge output because we don't have the edge annotation. then trained the model for 15 epochs with the binaryCrossEntropy Loss function on (74 + 34) images. We ended up with a very high accuracy and IOU score as we can see in table 10.

WBC Model	epochs	Loss	Mean IOU	Dice	Tversky	Accuracy
UNet	15	0.0120	0.0162	0.0863	0.0863	0.9976

Table 10: WBC model performance

After training the model we did a benchmark on the 13 images then on the complete database (108 images) to calculate the counting accuracy of the three methods.
The real count of the 108 images is calculated manually because the original database doesn't

PLT_Model	epochs	Loss	Mean IOU	Dice	Tversky	Accuracy
UNet	50	0.0156	0.1995	0.5085	0.5431	0.9946

Table 12: Platelets Model Performance

have the count information. We ended up with the table 11 below where we can see that the watershed method is the best for WBC counting with 97.94 accuracy on the 13 images and 95.64 on the complete database, because the white blood cells always slightly overlap each other where it's easy to the watershed to segment them.

Image	Real count	Watershed	CCL	CHT	Watershed_acc	CCL_acc	CHT_acc
Im037_0	1	1	2	1	100.00	0.00	100.00
Im045_0	1	1	2	1	100.00	0.00	100.00
Im053_1	45	44	39	41	97.78	86.67	91.11
Im001_1	18	17	14	15	94.44	77.78	83.33
Im004_1	12	12	10	14	100.00	83.33	83.33
Im015_1	22	22	16	20	100.00	72.73	90.91
Im022_1	5	5	6	5	100.00	80.00	100.00
Im050_1	21	22	18	24	95.24	85.71	85.71
Im069_0	1	1	2	1	100.00	0.00	100.00
Im079_0	7	8	8	6	85.71	85.71	85.71
Im095_0	2	2	3	2	100.00	50.00	100.00
Im099_0	3	3	4	1	100.00	66.67	33.33
Im108_0	4	4	5	4	100.00	75.00	100.00
Total 13 images					97.94	58.74	88.7
Total 108 images					95.64	51.68	85.76

Table 11: WBC Counting results using 3 algorithms

3.3 platelets

The platelets are easy to count because of the rare overlapping but they are a bit difficult to segment because of their small size.

In this experiment we are testing single output U-Net from [38]. In the U-Net we removed the edge output because we don't have the edge annotation. then trained the model for 50 epochs with the BinaryCrossEntropy Loss function on (74 + 34) images. We ended up with a very high accuracy and IOU score as we can see in table 12.

After training the model we did a benchmark on the 13 images to calculate the counting

accuracy of the three methods. We are comparing to the real count which is calculated by feeding the ground truth platelets masks to the CCL algorithm. because the original database doesn't have the count information.

We ended up with the table 13 below where we can see that the CCL method is the best for WBC counting with 98.58 accuracy, because of the rare overlapping on each other where it's easy to the CCL to segment them.

Image	Real count	CCL	CCL_acc
Im008_1	6	6	100.0
Im030_1	12	12	100.0
Im031_1	26	27	96.15
Im036_0	34	35	97.05
Im037_0	6	6	100.0
Im038_0	31	30	96.77
Im039_0	39	40	97.43
Im040_0	53	50	94.33
Im044_0	36	36	100.0
Im045_0	39	39	100.0
Im047_0	33	32	96.96
Im050_1	9	9	100.0
Im052_1	4	4	100.0
Im058_1	11	11	100.0
Im068_0	9	9	100.0
Total			98.58

Table 13: Platelets Counting results using Connected Component Labeling Algorithm

4. SegNet Results

SegNet segmentation results were pretty accurate for white blood cells and platelets, as for red blood cells, the segmentation was done using dual output (mask and edge-mask) to get rid of overlapped cells.

Here are the results of the Mean Squared Error (MSE) loss function on each type of cell:

SegNet	Output	Dataset	Epochs	Loss	Mean IOU	Dice	Tversky	Accuracy
Red Blood Cells	**Mask**	13	700	0.0315	0.7660	0.8902	0.9072	0.9586
	Edge			0.0436	0.0664	0.3618	0.3637	0.9399
White Blood Cells	**Mask**	106	80	0.0021	0.0214	0.2512	0.2572	0.9972
Platelets	**Mask**	102	80	0.0025	0.0001	0.0019	0.0031	0.9989

Table 14: Result of SegNet segmentation

4.1 Red Blood Cells

For the dual-output SegNet model, the resulting segmented images were very good, sometimes better than the do-U-Net, though it is not as optimized when training and also predicting images, but it gets the job done with 95.86% mask and 93.99% edge accuracies. The segmented output images also had some noise which affected Connected Component Labeling (CCL) when counting. As for Circle Hough Transform (CHT), the noise did not affect the result. Red Blood Cells detection and counting is by far the hardest, because it is the only cell that overlaps and that makes it hard for counting. The segmented output of do-SegNet is thresholded using a binary threshold, and then sent to 3 algorithms:

- **Circle Hough Transform**: CHT was our best result for red blood cells counting, which achieved an accuracy of 94.03% on the same dataset used for training the model (13 images with their respective masks and edge-masks).
- **Connected Component Labeling**: CCL was applied directly on the thresholded output edge, this method was far from accurate because the do-SegNet output had some noise (even when removing most of it), and also the overlapped nature of red blood cells which makes it very hard for this algorithm to count correctly. CCL achieved an accuracy of 76.49% counting red blood cells.
- **Euclidean Distance Transform**: EDT is used to get rid of the overlapped cells, also peak local max was applied on the EDT output for finding local maxima(s), the result of this approach is 84.64% accuracy.

Image	Real_Count	CHT	CCL	EDT	CHT_acc	CCL_acc	EDT_acc
Im001_1	215	212	234	247	98.6	91.16	85.12
Im004_1	258	255	264	285	98.84	97.67	89.53
Im015_1	293	269	212	261	91.81	72.35	89.08
Im022_1	242	232	279	268	95.87	84.71	89.26
Im037_0	105	103	75	88	98.1	71.43	83.81
Im045_0	601	576	409	480	95.84	68.05	79.87
Im050_1	460	486	450	475	94.35	97.83	96.74
Im053_1	802	614	228	448	76.56	28.43	55.86
Im069_0	665	653	376	549	98.2	56.54	82.56
Im079_0	512	502	407	430	98.05	79.49	83.98
Im095_0	150	177	177	170	82.0	82.0	86.67
Im099_0	528	516	437	465	97.73	82.77	88.07
Im108_0	510	528	418	458	96.47	81.96	89.8
Total					94.03	76.49	84.64

Table 15: RBC counting results using 3 algorithms

4.2 White Blood Cells

The results of white blood cells segmentation and counting using the SegNet model was very accurate achieving 99.72% when segmenting. White blood cells are the easiest out of the three, and the most accurate results. However, white blood cells are different from the other cells because they come in different shapes and sizes, which made it hard to adapt each counting algorithm to every cell. The same counting methods are applied CHT, CCL and EDT. And each method had some drawbacks.

Here are the results:

- **Circle Hough Transform**: Due to the different shapes of white blood cells, CHT achieved the lowest result which is 79.9% counting accuracy, because some of the cells don't even look like circles and also their different size which made it harder to count.
- **Connected Component Labeling**: CCL is similar to CHT when it comes to white

blood cells. And, because of the noisy outputs of SegNet the binary threshold can only do so much (it thresholds some of the noise generated when predicting).
CCl achieved an average counting accuracy of 82.89%.

- **Euclidean Distance Transform**: EDT is the contender of white blood cells counting, because the distance transform gets rid of the noise completely and peak local max was very helpful in eliminating that noise and getting an accurate count.
This method achieved a counting accuracy of 96.43%.

Here are the results of the 13 images:

Image	Real_Count	CHT	CCL	EDT	CHT_acc	CCL_acc	EDT_acc
Im001_1	18	13	13	16	72.22	72.22	88.89
Im004_1	12	11	9	12	91.67	75.0	100.0
Im015_1	22	18	15	21	81.82	68.18	95.45
Im022_1	5	5	5	5	100.0	100.0	100.0
Im037_0	1	0	1	1	0	100.0	100.0
Im045_0	1	1	1	1	100.0	100.0	100.0
Im050_1	21	29	20	21	61.9	95.24	100.0
Im053_1	45	44	43	44	97.78	95.56	97.78
Im069_0	1	1	1	1	100.0	100.0	100.0
Im079_0	7	7	9	9	100.0	71.43	71.43
Im095_0	2	2	2	2	100.0	100.0	100.0
Im099_0	3	1	3	3	33.33	100.0	100.0
Im108_0	4	4	8	4	100.0	0	100.0
Total					79.9	82.89	96.43

Table 16: WBC counting results using 3 algorithms

4.3 Platelets

The platelets segmentation result we achieved is not the best compared to the do-U-Net model. SegNet extracts the platelets but with some noise which made it very hard to count accurately. It achieved a segmentation accuracy of 99.89% and the highest counting accuracy

is 71.56% which is not very good.

Here are all the counting accuracies for each approach:

Image	Real_Count	CHT	CCL	EDT	CHT_acc	CCL_acc	EDT_acc
Im001_1	0	0	0	0	100.0	100.0	100.0
Im004_1	4	2	3	2	50.0	75.0	50.0
Im015_1	7	1	2	1	14.29	28.57	14.29
Im022_1	15	8	10	8	53.33	66.67	53.33
Im037_0	6	5	6	5	83.33	100.0	83.33
Im045_0	39	24	25	24	61.54	64.1	61.54
Im050_1	10	8	8	8	80.0	80.0	80.0
Im053_1	12	9	12	9	75.0	100.0	75.0
Im069_0	3	3	3	3	100.0	100.0	100.0
Im079_0	0	2	3	2	0.0	0.0	0.0
Im095_0	7	6	6	6	85.71	85.71	85.71
Im099_0	37	26	26	26	70.27	70.27	70.27
Im108_0	5	6	7	6	80.0	60.0	80.0
Total					65.65	71.56	65.65

Table 17: Platelets counting results using 3 algorithms

5. Comparative study

DO-U-Net showed better results compared to SegNet, because of the noise generated by SegNet, platelets were near impossible to count. However, SegNet outperformed DO-U-Net when it came to edge-mask segmentation.

Here are the results of both models in table 18:

Blood Cells / Model	Output	DO-U-Net				SegNet			
		Segmetation Accuracy %	Counting Accuracy %			Segmetation Accuracy %	Counting Accuracy %		
			CHT	CCL	Watershed		CHT	CCL	Watershed
Red Blood Cells	**Mask**	**97.16**	**95.36**	78.66	90.43	95.86	94.03	76.49	84.64
	Edge	93.85				**93.99**			
White Blood Cells	**Mask**	**99.76**	88.7	58.74	**97.94**	99.72	79.9	82.89	96.43
Platelets	**Mask**	99.46	X	**98.58**	X	**99.89**	65.65	71.56	65.65

Table 18: Comparison of DO-U-Net and SegNet results

6. Result samples

Here is the resulting output of each model on Im037.jpg (see figure 40), segmenting all the cells Red, White and platelets:

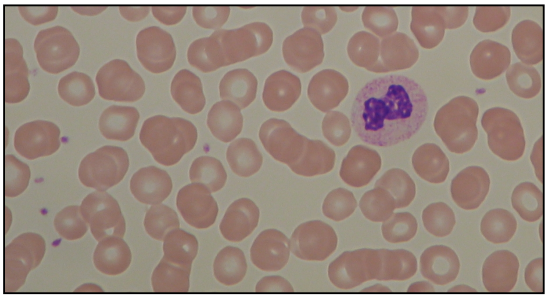
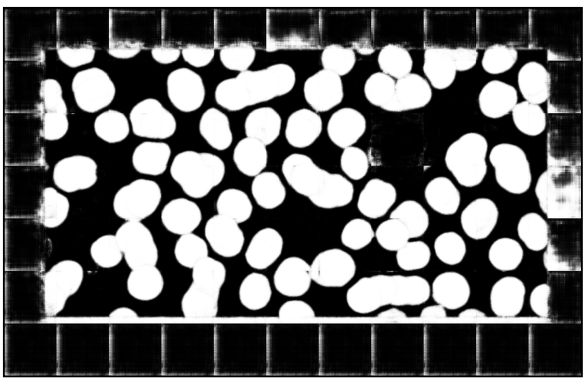
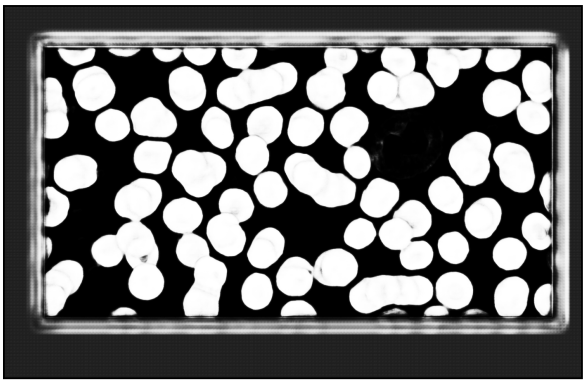


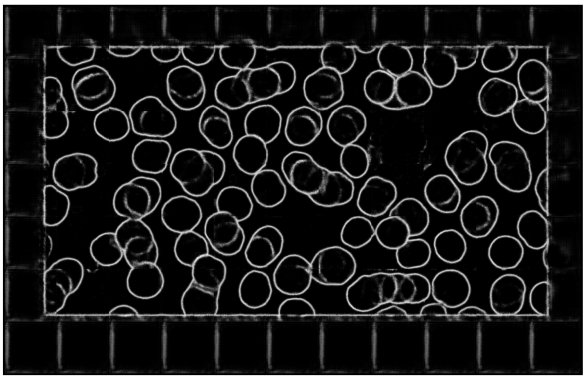
Figure 40: Original image Im037.jpg



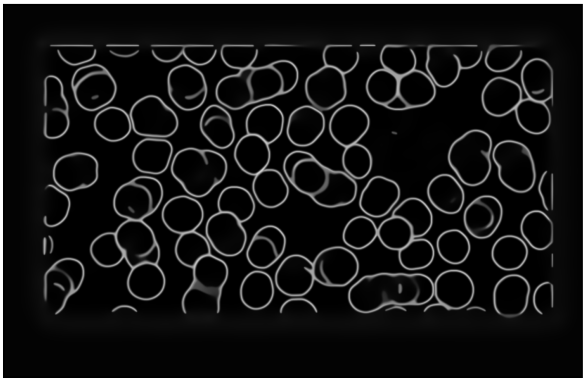
(a) Output SegNet mask



(b) Output DO-UNet mask

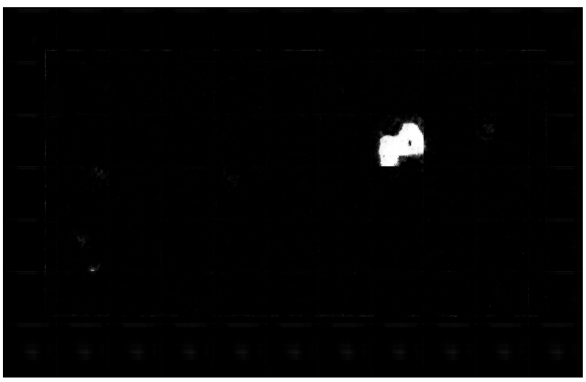


(c) Output SegNet edge

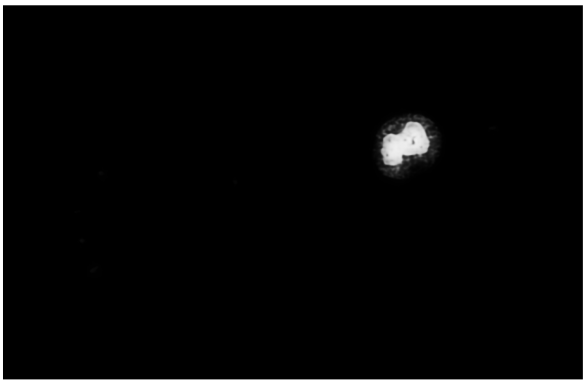


(d) Output DO-UNet edge

Figure 41: Output from SegNet and DO-UNet segmenting Red Blood Cells

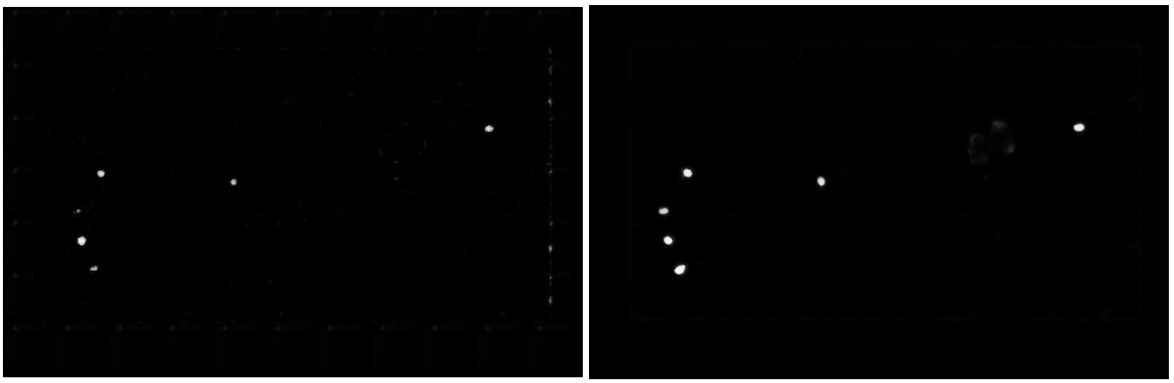


(a) Output white blood cells mask from SegNet



(b) Output white blood cells mask from UNet

Figure 42: Outputs from SegNet and UNet segmenting White Blood Cells

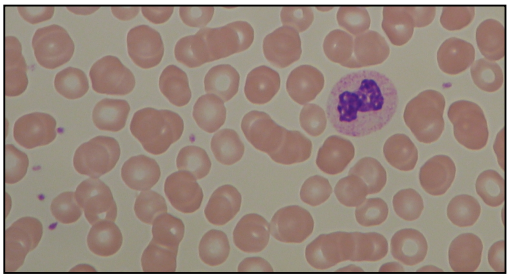


(a) Output mask of platelets with Segnet (b) Output mask of platelets with UNet

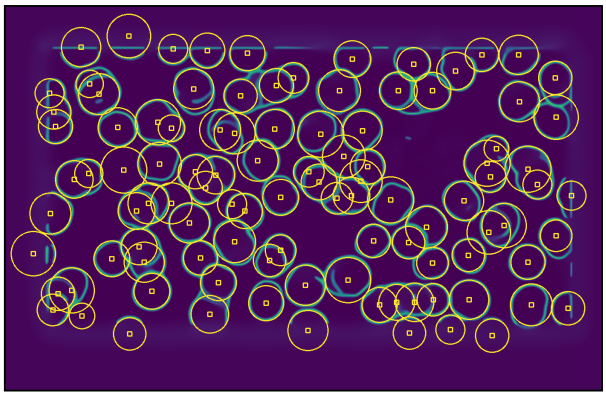
Figure 43: Outputs from SegNet and UNet segmenting platelets

After all the steps we have mentioned above, the final output of each counting algorithm is presented in the figures below:

- **Figure 44**: Circle Hough Transform applied on the segmented output of Red Blood Cells, the CHT method performed very well on the edge output of the do-U-Net model. Because of the circular shape of red blood cells and the parameter tuning of CHT, this is the best counting accuracy we achieved on Red Blood Cells.
- **Figure 45**: Watershed applied on the segmented output of White Blood Cells. Watershed outperformed the other algorithms because it segments touched and ovelapped objects. this is the best counting accuracy we achieved on White Blood Cells.
- **Figure 46**: Connected Component Labeling applied on the segmented output of White Blood Cells. This method performed well when it came to none touching objects (Platelets), in this figure also 2 Platelets where also detected as White Blood Cells, but they will get removed using the surface filter we implemented, which gets rid of small objects.

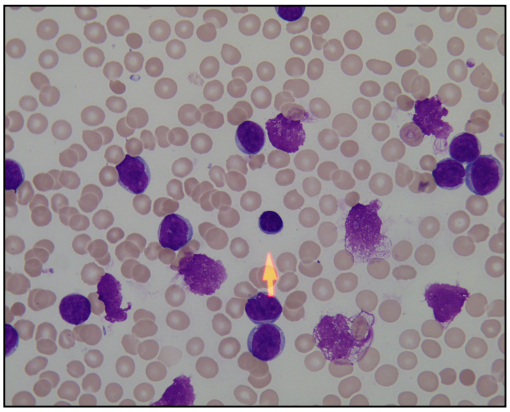


(a) Original image

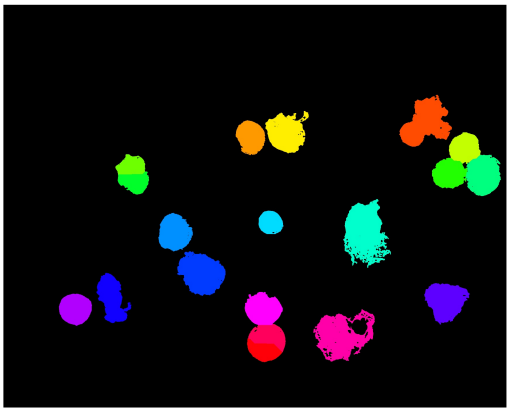


(b) Output CHT image

Figure 44: Circle Hough Transform applied to count red blood cells

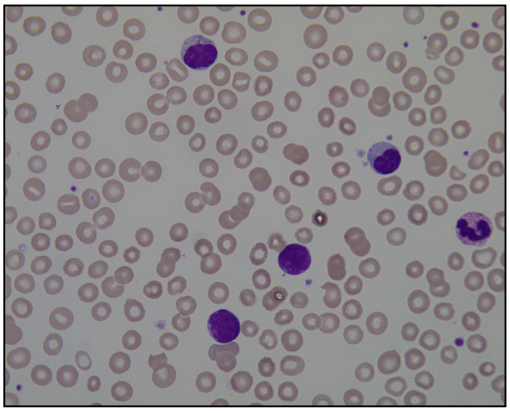


(a) Original image

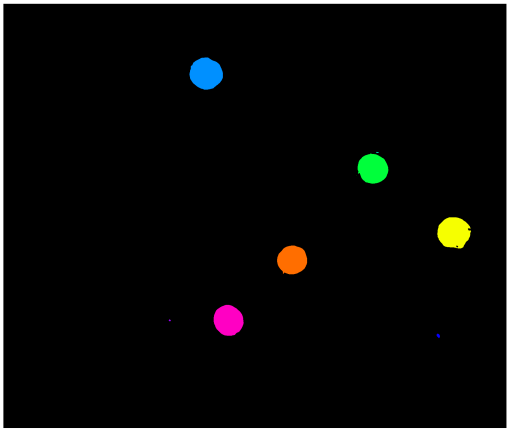


(b) Output Watershed image

Figure 45: Watershed applied to count white blood cells



(a) Original image



(b) Output CCL image

Figure 46: Connected Component Labeling applied to count white blood cells

7. Conclusion

In conclusion, both models performed very well detecting and counting the blood cells, each one with its advantages and disadvantages. But the DO-UNet model outperformed SegNet because of the noise generated using SegNet, especially when it comes to platelets since they are very similiar to that noise.

CONCLUSION

In the present work, we have mainly presented two different models of segmentation based on CNNs and three counting algorithms to perform a complete blood count. We focused more on the segmentation task where we tested two models U-Net and SegNet to get better masks.

We can see that the U-Net gave better result with less noisy mask, but the two models has some weaknesses with the images color space, because both models takes rgb images therefore they depend a lot on the color features, for example if we slightly change the colors of the input image we get a big diffrence in the output mask. In the counting task, we had a small time window where we couldn't tune the 3 three algorithm parameters to get the best results. but we got acceptable results in each blood cell type, especially with platelets.

As a future work, we can combine between the two models to benefit the strength points of both models, we can also find more combinations between edge and mask.
For the counting we can assign a weight for each method to get a mean value from the 3 methods. And it would be interesting to develop a system which makes it possible to classify each blood cell depending on the shape and size and color to detect even more sophisticated abnormalities more accurate than the complete blood count.

REFERENCES

[1] Nasir Ahmed, T_ Natarajan, and Kamisetty R Rao. Discrete cosine transform. *IEEE transactions on Computers*, 100(1):90–93, 1974.

[2] askanydifference. Difference between semi-supervised and reinforcement learning (with table) – ask any difference.
URL: https://askanydifference.com/difference-between-semi-supervised-and-reinforcement-learning/, 2022.

[3] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[4] Lata A Bhavnani, Udesang K Jaliya, and Mahasweta J Joshi. Segmentation and counting of wbcs and rbcs from microscopic blood sample images. *International Journal of Image, Graphics and Signal Processing*, 8(11):32, 2016.

[5] Gaudenz Boesch. Vgg very deep convolutional networks (vggnet) - what you need to know - viso.ai.
URL: https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/, 2022.

[6] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[7] cellavision. Cellavision news blast – cellavision ab.
URL: https://blog.cellavision.com/, 2022.

[8] Hao Chen, Xiaojuan Qi, Lequan Yu, and Pheng-Ann Heng. Dcan: deep contour-aware networks for accurate gland segmentation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2487–2496, 2016.

[9] clevelandclinic. White blood cells: What are they, normal ranges, role & function. *URL: https://my.clevelandclinic.org/health/body/21871-white-blood-cells*, 2022.

[10] H Digabel and C Lantuejoul. Iterative algorithms, actes du second symposium europeen d'analyse quantitative des microstructures en sciences des materiaux, biologie et medecine, caen, 4-7 october 1977, j.-l. chermant, ed. *Riederer Forlag, Stuttgart*, 1978.

[11] Steve Eddins. The watershed transform: Strategies for image segmentation - matlab & simulink.
URL: https://es.mathworks.com/company/newsletters/articles/the-watershed-transform-strategies-for-image-segmentation.html, 2022.

[12] Sarah F Frisken, Ronald N Perry, Alyn P Rockwood, and Thouis R Jones. Adaptively sampled distance fields: A general representation of shape for computer graphics. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 249–254, 2000.

[13] Kunihiko Fukushima, Sei Miyake, and Takayuki Ito. Neocognitron: A neural network model for a mechanism of visual pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(5):826–834, Sep. 1983.

[14] GeeksForGeeks. Supervised and unsupervised learning - geeksforgeeks.
URL: https://www.geeksforgeeks.org/supervised-unsupervised-learning/, 2022.

[15] FENG Guiliang, LU Yiping, and PENG Wei. Microscopic cell image segmentation and counting algorithm based on image definition. *International Journal of Simulation–Systems, Science & Technology*, 17(38), 2016.

[16] Carlos X. Hernández, Mohammad M. Sultan, and Vijay S. Pande. Using deep learning for segmentation and counting within microscopy data. *CoRR*, abs/1802.10548, 2018.

[17] Jean Hopkins, Anthea Maton, WM Charles, J Susan, QW Maryanna, L David, and DW Jill. Human biology and health. *New Jersey: Englewood Cliffs*, 1993.

[18] IBM. What are convolutional neural networks? | ibm.
URL: https://www.ibm.com/cloud/learn/convolutional-neural-networks, 2020.

[19] IBM. What is machine learning? | ibm.
URL: https://www.ibm.com/cloud/learn/machine-learning, 2022.

[20] Paul Jaccard. The distribution of the flora in the alpine zone. 1. *New phytologist*, 11(2):37–50, 1912.

[21] Vinod V Kimbahune and NJ Uke. Blood cell image segmentation and counting. *Inter J Engineer Sci Tech*, 3(3):2448–2453, 2011.

[22] Yan Kong, Hui Li, Yongyong Ren, Georgi Z. Genchev, Xiaolei Wang, Hongyu Zhao, Zhiping Xie, and Hui Lu. Automated yeast cells segmentation and counting using a parallel u-net based two-stage framework. *OSA Continuum*, 3(4):982–992, Apr 2020.

[23] Zahra Mousavi Kouzehkanan, Sepehr Saghari, Sajad Tavakoli, Peyman Rostami, Mohammadjavad Abaszadeh, Farzaneh Mirzadeh, Esmaeil Shahabi Satlsar, Maryam Gheidishahran, Fatemeh Gorgi, Saeed Mohammadi, and Reshad Hosseini. A large dataset of white blood cells containing cell locations and types, along with segmented nuclei and cytoplasm. *Scientific Reports*, 12(1):1123, Jan 2022.

[24] Ruggero Donida Labati, Vincenzo Piuri, and Fabio Scotti. All-idb: The acute lymphoblastic leukemia image database for image processing. In *2011 18th IEEE international conference on image processing*, pages 2045–2048. IEEE, 2011.

[25] Dongming Li, Peng Tang, Run Zhang, Changming Sun, Yong Li, Jingning Qian, Yan Liang, Jinhua Yang, and Lijuan Zhang. Robust blood cell image segmentation method based on neural ordinary differential equations. *Computational and Mathematical Methods in Medicine*, 2021, 2021.

[26] A Rectified Linear. U-net explained | papers with code.
URL: https://paperswithcode.com/method/u-net, 2022.

[27] Vebjorn Ljosa, Katherine L Sokolnicki, and Anne E Carpenter. Annotated high-throughput microscopy image sets for validation. *Nature methods*, 9(7):637–637, 2012.

[28] LR-GeeksForGeeks. Ml | linear regression - geeksforgeeks.
URL: https://www.geeksforgeeks.org/ml-linear-regression/, 2022.

[29] K. Malarz, S. Kaczanowska, and K. Kulakowski. Chaotic dynamics of forest fires.
URL: https://arxiv.org/abs/cond-mat/0204492, 2002.

[30] Carsten Maple. Geometric design and space planning using the marching squares and marching cube algorithms. In *2003 international conference on geometric modeling and graphics, 2003. Proceedings*, pages 90–95. IEEE, 2003.

[31] mayoclinic. Complete blood count (cbc) - mayo clinic.
URL: https://www.mayoclinic.org/tests-procedures/complete-blood-count/about/pac-20384919, 2022.

[32] John McCarthy. What is artificial intelligence.
URL: http://www-formal.stanford.edu/jmc/whatisai.html, 2004.

[33] Shervin Minaee, Yuri Y Boykov, Fatih Porikli, Antonio J Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2021.

[34] ml cheatsheet. Gradient descent — ml glossary documentation.
URL: https://ml-cheatsheet.readthedocs.io/en/latest/gradient_descent.html, 2022.

[35] Allan H Murphy. The finley affair: A signal event in the history of forecast verification. *Weather and forecasting*, 11(1):3–20, 1996.

[36] William S Noble. What is a support vector machine? *Nature biotechnology*, 24(12):1565–1567, 2006.

[37] Alan T Nurden, Paquita Nurden, Mikel Sanchez, Isabel Andia, Eduardo Anitua, et al. Platelets and wound healing. *Front Biosci*, 13(9):3532–48, 2008.

[38] Toyah Overton and Allan Tucker. Do-u-net for segmentation and counting. In Michael R. Berthold, Ad Feelders, and Georg Krempl, editors, *Advances in Intelligent Data Analysis XVIII*, pages 391–403, Cham, 2020. Springer International Publishing.

[39] Simon Just Kjeldgaard Pedersen. Circular hough transform. *Aalborg University, Vision, Graphics, and Interactive Systems*, 123(6), 2007.

[40] URMC rochester. What are red blood cells? - health encyclopedia - university of rochester medical center.
URL: https://www.urmc.rochester.edu/encyclopedia/content.aspx?ContentID=34&ContentTypeID=160, 2022.

[41] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[42] Omid Sarrafzadeh, Hossein Rabbani, Ardeshir Talebi, and Hossein Usefi Banaem. Selection of the best features for leukocytes classification in blood smear microscopic images. In *Medical Imaging 2014: Digital Pathology*, volume 9041, pages 159–166. SPIE, 2014.

[43] Fabio Scotti, Ruggero Donida Labati, and Prof. Vincenzo Piuri. All-idb (acute lymphoblastic leukaemia-international database).
URL: https://dx.doi.org/10.21227/pm77-2n23, 2020.

[44] Muhammad Shahzad, Arif Iqbal Umar, Muazzam A Khan, Syed Hamad Shirazi, Zakir Khan, and Waqas Yousaf. Robust method for semantic segmentation of whole-slide blood cell microscopic images. *Computational and Mathematical Methods in Medicine*, 2020, 2020.

[45] Shenggan. Bccd dataset.
URL: https://github.com/Shenggan/BCCD_Dataset, 2017.

[46] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[47] K. Sudha and P. Geetha. A novel approach for segmentation and counting of overlapped leukocytes in microscopic blood images. *Biocybernetics and Biomedical Engineering*, 40(2):639–648, 2020.

[48] tecom cn. Jiangxitecom biochemistry equipment co., ltd., hematology analyzers biochemistry analyzers reagents.
URL: https://en.tecom-cn.com/, 2022.

[49] Thanh Tran, Lam Binh Minh, Suk-Hwan Lee, and Ki-Ryong Kwon. Blood cell count using deep learning semantic segmentation. 2019.

[50] Vincent and Soille. Watershed segmentation - an overview | sciencedirect topics. *URL: https://www.sciencedirect.com/topics/computer-science/watershed-segmentation*, 2022.

[51] Annie Stuart What. Acute lymphoblastic leukemia (all): Symptoms, diagnosis, treatment, prognosis, and survival rate. *URL: https://www.webmd.com/cancer/lymphoma/acute-lymphoblastic-leukemia*, 2022.

[52] Wikipedia. Circle hough transform - wikipedia. *URL: https://en.wikipedia.org/wiki/Circle_Hough_Transform*, 2022.

[53] Xin Zheng, Yong Wang, Guoyou Wang, and Jianguo Liu. Fast and robust segmentation of white blood cell images by self-supervised learning. *Micron*, 107:55–71, 2018.