University of Louisville

ThinkIR: The University of Louisville's Institutional Repository

Electronic Theses and Dissertations

5-2020

Computational techniques in medical image analysis application for white blood cells classification.

Omar Dekhil
University of Louisville

Follow this and additional works at: https://ir.library.louisville.edu/etd

Part of the Other Computer Engineering Commons

Recommended Citation
Dekhil, Omar, "Computational techniques in medical image analysis application for white blood cells classification." (2020). *Electronic Theses and Dissertations.* Paper 3424.
Retrieved from https://ir.library.louisville.edu/etd/3424

This Doctoral Dissertation is brought to you for free and open access by ThinkIR: The University of Louisville's Institutional Repository. It has been accepted for inclusion in Electronic Theses and Dissertations by an authorized administrator of ThinkIR: The University of Louisville's Institutional Repository. This title appears here courtesy of the author, who has retained all other copyrights. For more information, please contact thinkir@louisville.edu.

COMPUTATIONAL TECHNIQUES IN MEDICAL IMAGE ANALYSIS: APPLICATION FOR WHITE BLOOD CELLS CLASSIFICATION

By
Omar Dekhil

A Dissertation
Submitted to the
J.B. Speed School of Engineeringof the University of
Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy
in Computer Science and Engineering

Department of Computer Science and Engineering
University of Louisville
Louisville, Kentucky

May 2020

COMPUTATIONAL TECHNIQUES IN MEDICAL IMAGE ANALYSIS:
APPLICATION FOR WHITE BLOOD CELLS CLASSIFICATION

By

Omar Dekhil

Dissertation approved on

April 15, 2020

by the following Dissertation Committee:

Dissertation Director
Dr. Adel Elmaghraby

Dr. Ibrahim Imam

Dr. Hui Zhang

Dr. Daniel Sierrasosa

Dr. Monica Gentili

DEDICATION

I dedicate my dissertation work to my family. A special feeling of gratitude to my loving parents whose words of encouragement and push for tenacity ring in my ears. My wife and my daughter, who have never left my side and helped me a lot.

ACKNOWLEDGMENTS

I wish to thank my committee members who were more than generous with their expertise and precious time. A special thanks to Dr. Adel Elmaghraby, my committee chairman for his countless hours of reflecting, reading, encouraging, and most of all patience throughout the entire process. Thank you Dr. Ibrahim Imam, Dr. Hui Zhang, Dr. Daniel Sierrasosa , and Dr. Monica Gentili for agreeing to serve on my committee.

ABSTRACT

COMPUTATIONAL TECHNIQUES IN MEDICAL IMAGE ANALYSIS:
APPLICATION FOR WHITE BLOOD CELLS CLASSIFICATION

Omar Dekhil

April 15,2020

White blood cells play important rule in the human body immunity and any change in their count may cause serious diseases. In this study, a system is introduced for white blood cells localization and classification. The dataset used in this study is formed by two components, the first is the annotation dataset that will be used in the localization (364 images), and the second is labeled classes that will be used in the classification (12,444 images). For the localization, two approaches will be discussed, a classical approach and a deep learning based approach. For the classification, 5 different deep learning architectures will be discussed, 3 pretrained architectures and 2 customized architectures will be presented.

After discussing this models and test them on the dataset, the best selected model will be evaluated describing the obtained results. The localization module achieved average Intersection over Union (IoU) of 71%, while the classification module achieved 92 % classification accuracy. In addition to reporting the model performance, the model robustness was also checked by adding three different types of noise, Gaussian noise, salt and pepper noise, and speckle noise.

This system outperforms other studies in the literature, where the accuracy was either less than the obtained from the system or the dataset was much smaller the used data in this study.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Automating medical computer aided diagnosis (CAD) systems is one of the most hot topics of research [7]. There have been enamours efforts to develop CAD systems for different medical applications. For example, several studies [8, 9, 10] suggested using mammography images to build CAD systems for breast cancer detection and classification. Another studies, for example [11, 12, 13] used computed tomography (CT) to build CAD systems for classification and detection of lung cancer. Another CAD systems were developed for different cancer types. For prostate cancer, different studies [14, 15, 16] used MRI to build CAD systems for classification and diagnosis.

In addition to cancer diagnosis, CAD systems were used for different medical applications. One of these applications is the diabetic retinopathy grade assessment. In this application, OCT images are the most commonly used [17, 18, 19]. Another application where CAD systems were used, is the ulcer detection. In many studies capsule endoscopy [20, 21] imaging was used. Moreover, some studies used images obtained by cell phone cameras to develop a more convenient CAD system for ulcer detection [22].

In this study, A CAD system is proposed and provided for white blood cells (WBC) classification and localization using microscopic images. Using the microscopic imaging could help providing important information regarding patient health status. A wide range of hermetic pathology could be revealed using differential blood count. For example, the presence of infections, leukemia, and some particular kinds of cancers can be diagnosed based on the results of the classification and the count of white blood cells. The traditional way to do this test is using a manual operators, where They use a microscope and count the percentage of the occurrence of each cell type counted within the region of interest in smears. Obviously, this manual counting process is very tedious and slow [23]. In the next sections, the background about WBCs and related disorders, then a literature review covering the previous related work for automating the WBCs classification will be discussed.

1.1 Background

White blood cells could be divided into 4 categories [23]:

1. Neutrophils: They are type of white blood cell (WBC or granulocyte) that are responsible to protect humans from infections. They represent approximately 40 to 60 percent of the white blood cells in our bodies (Figure 1 a).

2. Lymphocytes: They are a type of immune cell made in the bone marrow and found in the blood and in lymph tissue. There are two main types of lymphocytes cells: B lymphocytes and T lymphocytes. B lymphocytes are responsible for making antibodies, and T lymphocytes are responsible to kill tumor cells and help control immune responses. (Figure 1 b).

3. Eosinophils: They are a specialized cell of the immune system. This proinflammatory white blood cell generally has a nucleus with two lobes (bilobed) and cytoplasm filled with approximately 200 large granules containing enzymes and proteins with different (known and unknown) functions (Figure 1 c).

4. Monocytes: They are a type of white blood cell that fights off bacteria, viruses and fungi. Monocytes are the biggest type of white blood cell in the immune

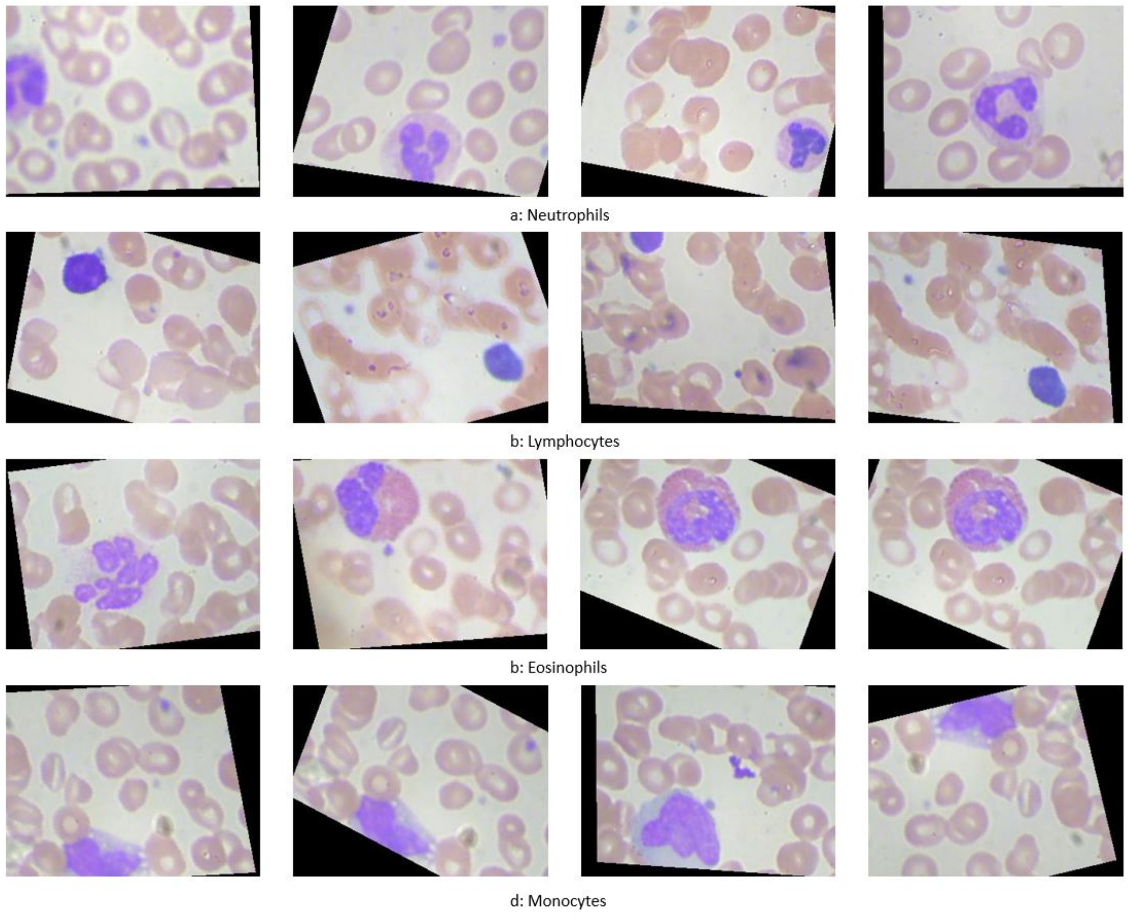

Figure 1. Examples of the four types of the WBCs, (a) Neutrophils. (b) Lymphocytes, (c) Eosinophils, and (d) Monocytes

system. Originally formed in the bone marrow, they are released into our blood and tissues. When certain germs enter the body, they quickly rush to the site for attack (Figure 1 d).

Figure 2 shows the 3D rendering of the 4 WBCs types.

The WBC disorders could be categorized into two main categories: (i) proliferative disorders, which are associated with an increment in WBCs, and (ii) leukopenias, which are associated with a decrease in WBCs. The most common WBC disorders include:

1. Leukocytosis: It is associated with an increased number of white blood cells. The most common causes include bacterial or viral infections, certain medications, allergies, smoking, inflammatory diseases, autoimmune disorders, a genetic condition, and cancer [24]

2. Autoimmune neutropenia: The main cause is producing antibodies that attack and destroy neutrophils. The condition is associated with various diseases, including Crohn's disease and rheumatoid arthritis [25].

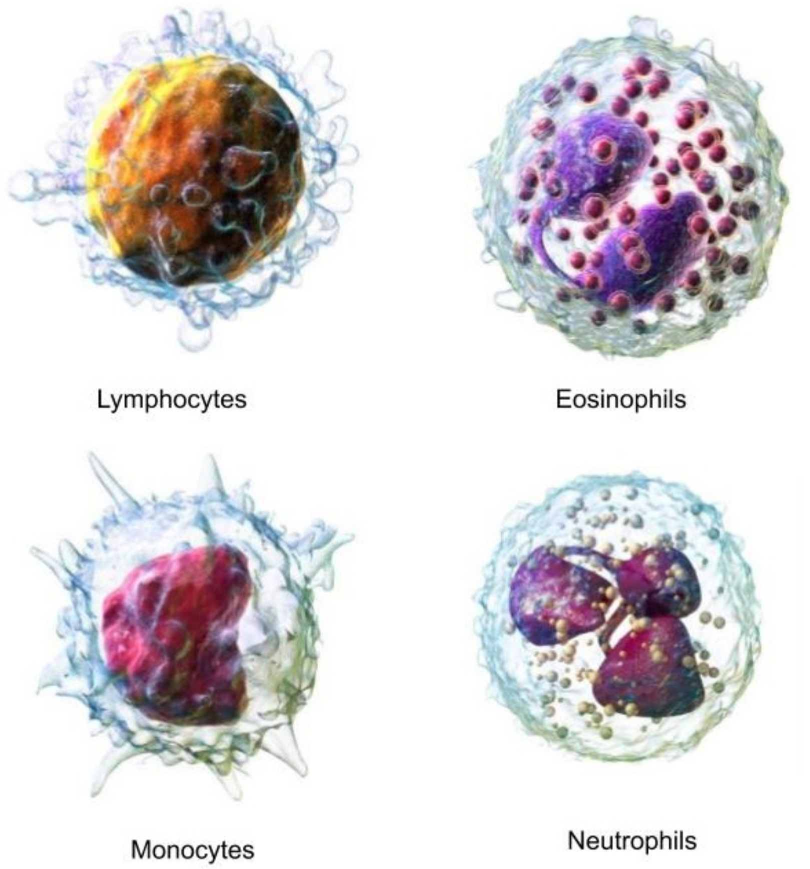

Figure 2. The 3D rendering of the 4 types of the white blood cells [2]

Table 1. The normal count of Neutrophils, Lymphocytes, Eosinophils, Monocytes per microliter

Type	**Normal range**
Neutrophils	1,800-8,300 cells/mcL
Lymphocytes	800-5,000 cells/mcL
Eosinophils	0-800 cells/mcL
Monocytes	400-1,000 cells/mcL

3. Severe congenital neutropenia: Which is caused by genetic mutation. Patients with severe congenital neutropenia have recurrent bacterial infections [26].

4. Cyclic neutropenia: It is also associated with genetic mutation. The neutropenia occurs in cycles of about 21 days [27].

5. Chronic granulomatous disease: This is a disorder where multiple types of WBCs (neutrophils, monocytes, macrophages) are not functioning properly. It is an inherited condition and it is associated with multiple infections, particularly pneumonia and abscesses [28].

6. Leukocyte adhesion deficiencies (LAD syndrome): It is arer genetic disorder where WBCs are unable to reach the infected areas [29].

The white blood cell count is usually obtained through Complete Blood Count (CBC). This test is used to get the count of red blood cells, white blood cells, hematocrit, and Platelets. The normal blood cells count is between 3400 and 9600 cells/microliter [30]. The normal range of each of the four types of WBCs is shown in Table 1

For the manual identification of the WBCs different types, the following shape characters are used in differentiating between them [31]:

1. In the microscopic images, neutrophils appear spherical in shape with a dark stained nucleus that is divided into 2 to 5 lobes. A closer look shows fine granules (neutrophilic granules) and thin threads connecting the nucleus lobes (chromatin threads)

2. Eosinophils: They only have a bi-lobed (two lobes) nucleus that is shaped like a horse-shoe. They also appear in spherical shape with fine granules called acidophilic refractive granules.

3. Lymphocytes : They are smaller compared to other leukocytes. They also have a large round nucleus that takes up much of the cell volume. As a result, lymphocytes have very little to no cytoplasm.

4. Monocytes: They larger in size than Lymphocytes with a nucleus that is bean or kidney shaped. These cells also have more cytoplasm compared to lymphocytes.

1.2 Literature review

1.2.1 State of the art in WBC localization, detection, and classification

Many previous studies addressed the problem of localization and classification of the WBC in microscopic images. The studies could be divided into two main categories: (i) using classical computer vision and machine learning techniques, and (ii) using deep learning techniques.

1.2.1.1 WBC localization and classification using classical computer vision and machine learning

In order to classify the WBC, Ongun *et al.* [32] proposed a system with 3 steps: (i) WBC segmentation, (ii) feature extraction, and (iii) classification. For segmentation, they used active contours method followed by morphological operators. For the feature selection, they used 2 sets of features: (i) shape based features and (ii) texture and color based features. For the shape based features, selected features were the area of cell and nucleus, ratio of nucleas area and perimeter length over cell, energy of the nucleus, and nucleus shape features. For the texture color based features, affine invariant features and distance between colors were selected. For the classification, they reported the results using K-Nearest Neighbor (KNN), Linear vector Quantization (LVQ) , Multi Layer Perceptron (MLP), and Support vector Machine (SVM). The best achieved testing accuracy was 91.03% using SVM.

In another study [33], LSinha and Ramakrishnan proposed a framework for segmentaion, feature extraction, and classification. For the segmentation, K-means clustering was used to locate the WBC nuclei and extract the Region of Interest (ROI). After extracting the ROI, Expectation maximization was then used to segment the cytoplasm and the nucleus. After the segmentation, three types of features were extracted, the color features, the shape features, and the texture features. The selected color feature was the the mean value for each color value. For the texture features, energy, entropy, and auto-correlation matrix were used. In the classification phase, Neural Networks (NN), KNN, Bayes classifier, and SVM were tested. The highest achieved accuracy was 94.1% with a dataset of 50 training samples and 34 testing samples.

In [34], an algorithm for segmenting WBC using wavelet transform was proposed. The proposed algorithm in [34] was to apply wavelet transform to the original images then the low frequency components and high frequency components separately. For the high frequency components, soft threshold is applied, followed by wavelet coefficient threshold, while for the low frequency components, watershed segmentation is applied, followed by region combination. After processing both components, inverse wavelet transform is applied to obtain the segmented images.

A segmentation technique was suggested in [35] that uses neutrosophic similarity [36] score to measure the similarity between different components of the blood smear image. Since different color components from different color spaces were utilized in this approach, the neutrosphic score algorithm to was modified be adaptive.

In [37], several image processing techniques were combined to segment WBCs. The suggested pipeline in [37] could be divided into two main parts: (i) segmenting nucleus, and (ii) segmenting cytoplasm. To segment the nucleus, the following steps are applied.

1. Converting image to gray scale
2. Extracting WBC region of interest.
3. Detecting edges.
4. Segmenting nuclei using GVF snake [38]
5. Using morphological operators for hole filling.

Once the nucleus is segmented, it is subtracted from the gray scale images, then Zack thresholding [39] is used to segment cytoplasm. This approach resulted in 92% accuracy for nucleus segmentation and 78% for cytoplasm segmentation.

In a recent study [40], a framework for nucleus segmentation and classification was proposed. For the segmentation, the color image is first separated into the red, green, and, blue channels, the contrast of the green channel is then enhanced, the red channel is then replaced with contrast enhanced green channel to form GGB image, TissueQuant algorithm is then applied to the image, image is transformed to binary image using thresholding, and finally, filter of 8000 pixels and morphological closing of disk shaped structuring element of size 10 to obtain the normalized black and white image, I_{bwN}, is applied. For the classification, shape features and texture features were used. The selected shape features were area, perimeter, circularity, convexity and solidity, while the selected texture features were mean, variance, skewness and kurtosis, and Spatial Gray Level Dependence Matrix (SGLDM). For the classification, three stages classifier were used, the first stage is SVM classifier to detect Basophils, then another SVM to detect Lymphocytes, and finally neural network to decaffeinate between Monocyte, Neutrophils, and Eosinophils. The average obtained accuracy using this approach is 96.64 when tested on 117 images.

1.2.1.2 WBC localization and classification using deep learning

Many studies tried to use deep learning techniques for the WBc classification task. For example, in [41], a pipeline was proposed that uses deep learning for WBCs classification. The proposed approach in [41] consists of 4 main stages: (i) preprocessing, (ii) feature extraction, (iii) feature selection, and (iv) classification. The preprocessing step aims mainly to adjust the size of the input images to match the size of the pretrained network input. For the feature extraction, several pretrained networks were tested like OverfeatNet, AlexNet, VGG, in addition, a novel archtitecture trained from scratch was also tested. In the feature selection, chi-squared algorithm was used. Finally, SVm was used for the classification. The highest achieved accuracy was 96.1% using the novel architecture. This accuracy was obtained on a dataset containing 2551 images.

In a recent study [42], a CAD system was proposed. This CAD system has two main stages, detection and classification. For the detection Single Shot Detector (SSD) [43] was used, while for the classification different pretrained networks were tested. The used architectures were AlexNet [44], VGG, GoogleNet[6], and ResNet. The highest achieved average accuracy was 0.97 using AlexNEt when tested on 7500 images.

In [45], a Double Convolution Layer Neural Network (DCLNN) architecture was proposed. This architecture is relatively simple when compared to the pretrained architectures, and it is trained from scratch. This architecture contains two convolution layers, each of them is followed by a pooing layer. After the second pooling layer, there a fully connected later, then the output layer. This approach achieved average classification accuracy of 88% when tested on a dataset containing 13,000 images.

Another recent study [46], proposed a CAD system for WBCs classification using capsule networks. The main idea in capsule networks is to add structures called “capsules” to the CNN. The output of these capsules is used to form more stable representations for higher level capsules. The output of the capsule networks is the probability of an observation, in addition there a pose output for each observation [47]. The architecture of the capsule networks consist mainly of decoder and encoder. The encoder is responsible for encoding the image to 16 dimensional vector which contains the information needed tor render the image. The decoder is used to learn how to decode the instantiation parameters given into an image of the object it is detecting. The decoder uses the Euclidean distance in its loss function to measure the similarity between the reconstructed feature and the feature that it is being trained on. The main advantage when using the capsule networks is having reconstructed images that could be visually inspected. Using this architecture, the average obtained accuracy was 92.5% when tested on dataset having 263 images.

In [48] a deep learning architecture was proposed to segment white blood cells and red blood cells. The proposed architecture in [48] was SegNEt architecture[49]. This architecture uses pairs of encoder and decoder to create feature maps for segmentation. For the Encoders used in SegNet, they are formed of convolutional layer, batch normalization and with ReLu activation function. To reduce the feature maps size, maxpooling is used. As a result of maxpooling, the edges are blurred, that is why it is important to store the edges information in the encoder feature maps. At the decoder stage, it is important to restore the same input size. That is why the decoder includes upsampling layers and memorized max-pooling indices obtained from the encoder’s feature maps. With this architecture, a dataset of 42 images was segmented, the mean intersection over union obtained was 79%.

1.3 How is this thesis organized?

After discussing the background about the white blood cells types and the literature review about localization, segmentation and classification of white blood cells, the next chapter will describe the method used. It will cover a detailed discussion about convolutional neural networks, pretrained networks and customized models. It will also discuss the Region of Interest (ROI) extraction using classical image processing technique and a deep learning based algorithm for the ROI extraction. In chapter 3, the experimental results will be discussed, the experiments will cover the classification for the whole images using pretrained networks and using customized models, it will also show the experimental results for ROI extraction and the effect of ROI extraction on the classification results. In addition, the effect of adding noise on the testing dataset will be also discussed to show the system robustness. Finally, in chapter 4, the conclusion, discussion, and future work will be discussed.

CHAPTER 2
METHODS

In this chapter, the used methods in the experiments are discussed and explained. The methods are divided into three main sections:

1. Pretrained weighs models (transfer learning): In this section, four of the widely used convolutional neural networks (CNNs) for classification will be discussed.

2. Customized developed deep learning models: In this section, two CNN architectures developed to classify the WBC data will be explained and discussed.

3. Region of interest extraction: In this section, we will discuss several methods for extracting ROI defining the white blood cells in the images. This will used in the experiments to evaluate the models using the whole images versus using the ROI cropped images only.

2.1 Pretrained networks and transfer learning

2.1.1 Background of transfer learning

Since the idea of transfer learning started to rise at the last century, transfer learning (also called cross-domain learning, or domain adaptation), has been widely studied as machine learning technique [50].

Recently, with the huge amount of available data on the internet in different domains (audio, text, images, and videos), and with the target tasks towards achieving higher accuracies, data scales, and computational efficiencies, transfer learning techniques started to gain increasing importance in different research areas, specially, in the domain of pattern recognition and machine learning [51].

In the computer vision, specially the image classification tasks, the classification of the ImageNet dataset [3] gained great importance. The ImageNet classification in not now used as end goal, but rather, it is used as an elementary step towards training deep CNNs for other classification tasks on different datasets [3].

ImageNet [3] is a huge image dataset designed for visual object recognition tasks. It contains more than 14 million manually annotated images. In addition to indicating what object is in the image, at least one million of the images contain bounding boxes

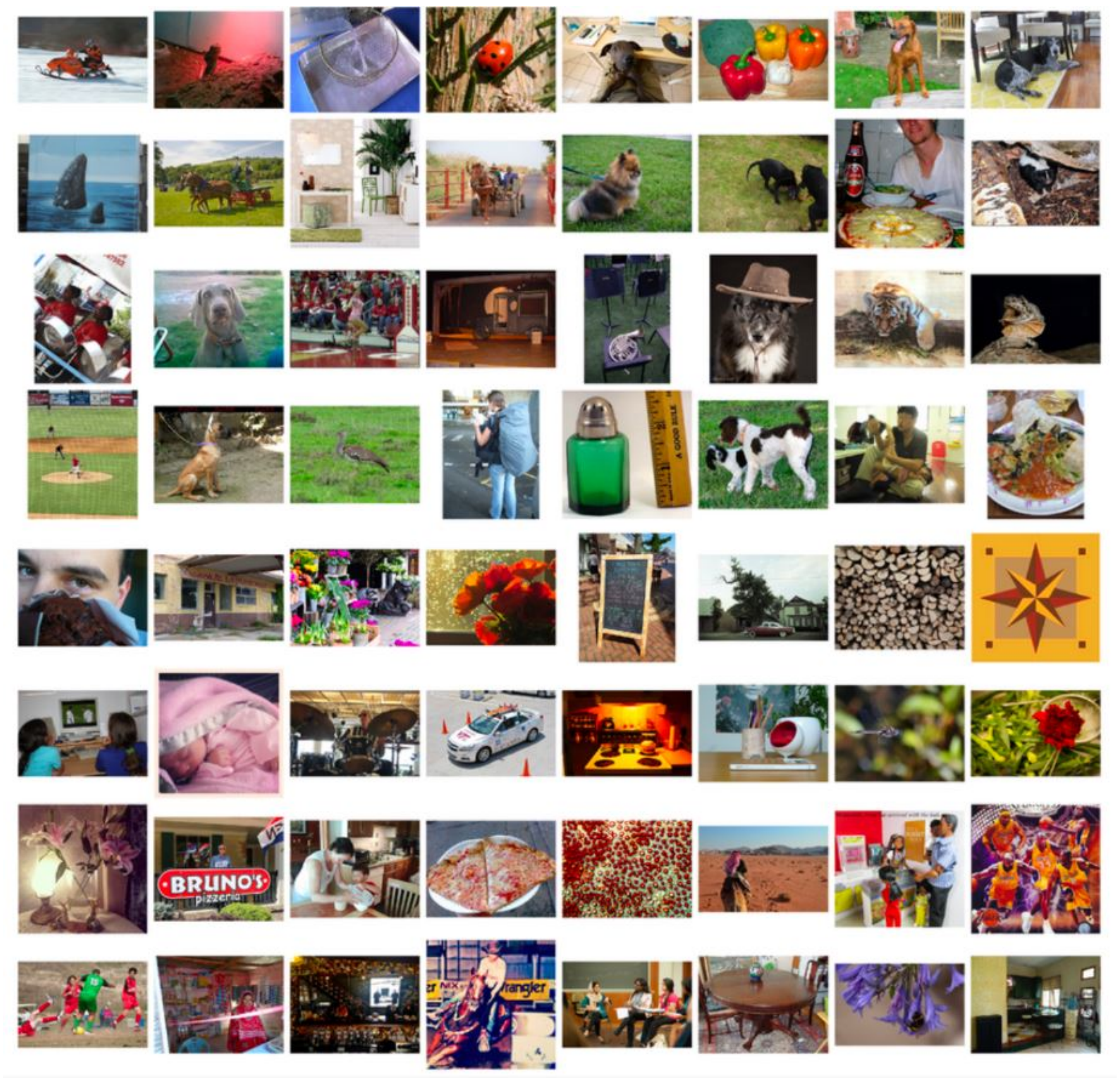

Figure 3. Sample of the images provided by the ImageNet dataset [3]

around the main objects in the images. The data in ImageNet has more than 20,000 categories. Figure 3 shows sample of the images presented in the dataset [3].

What makes the idea of transfer learning very appealing in the visual recognition tasks is the idea of weight sharing and higher level feature representation. In the different CNN's architectures, the first layers learn very basic feature like edges and corner. At the deeper layers networks start to learn higher level features (cars, human faces,etc.). This idea is very interesting in transfer learning as many objects share the same features learnt at one dataset. Figure 4 shows sample of the feature maps extracted from a face recognition CNN at different levels of the network [52].

In the next sub section, we will discuss the basic layers of CNN layers, then we will discuss four widely used CNN architectures for transfer learning:

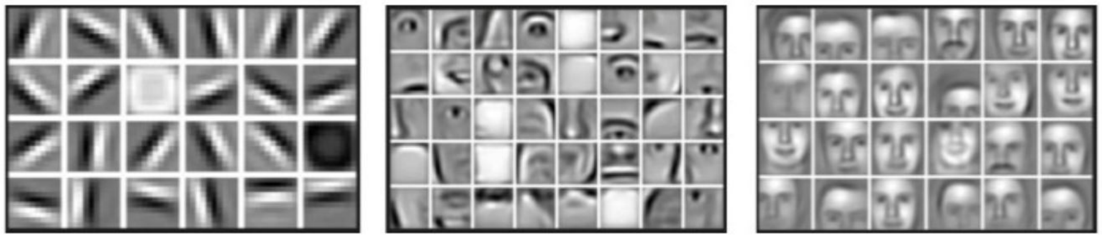

Figure 4. Sample of the feature maps generated by the CNN. At the first few layers, basic shapes are detected, and the deeper layers, higher level features are learnt

1. LeNet architecture
2. VGG architecture
3. ResNet architecture
4. Inception architecture

2.1.2 CNN layers

The main components of the CNN could be categorized into 4 types of layers:

1. Convolution layer: In this layer, multiple filters are applied to the input image. These filters generate feature maps that carries higher level features representation as we go deeper in the network. After each convolution layer, nonlinear activation is applied. The most widely used activation is ReLu activation [53].
2. Pooling layer: CNN may include local or global pooling layers. Pooling layers are used to reduce the dimensions of the data by combining the outputs of multiple neuron at one layer into one neuron in the deeper layers. Pooling may compute the maximum or the average over the block of neurons [54].
3. Batch normalization layer: Batch Normalization allows for using higher learning rates and be less careful about initialization. This allows for faster conversion times during the training phase. It also acts as a regularizer in some cases due its effect in reducing overfitting [55].
4. Dense layer: A dense layer is regular layer of neurons in the CNN. at the dense layer, each neuron receives input from all other neurons in the previous layer, thus densely connected.

2.1.3 LeNet Architecture

LeNet is one of the earliest architectures developed for image classification tasks [4]. This architecture is very simple. The LeNet architecture was designed for hand written digit recognition [56]

The LeNet architecture contains 6 layers:

1. First layer: The input for LeNet is a 32×32 grayscale image which passes through the first convolutional layer. In the first layer, 6 feature maps are generated using 5×5 convolutional filter with a stride of one. The output of this layer is 6 feature maps of size 28×28.
2. Second Layer: In this layer average pooling with a filter size 2×2 and a stride of two is applied. The resulting image dimensions will be reduced to$14 \times 14 \times 6$.
3. Third layer: This layer is the second convolution layer. In this layer, 16 feature maps of size 5×5 of stride one are used.
4. Fourth Layer: In this layer average pooling with a filter size 2×2 and a stride of two is applied. The resulting image dimensions will be reduced to $5 \times 5 \times 5$.
5. Fifth layer: The fifth layer is a fully connected convolutional layer with 120 feature maps each of size 1×1. Each of the 120 units in this layer is connected to all the 400 nodes in the previous layer.
6. Sixth layer: This layer is a fully connected layer with 84 units.

The output layer of the LeNet is a fully connected softmax output layer $\hat{y}$ with 10 possible values corresponding to the digits from 0 to 9. Figure 5 illustrates the architecture of LeNet as described in [4].

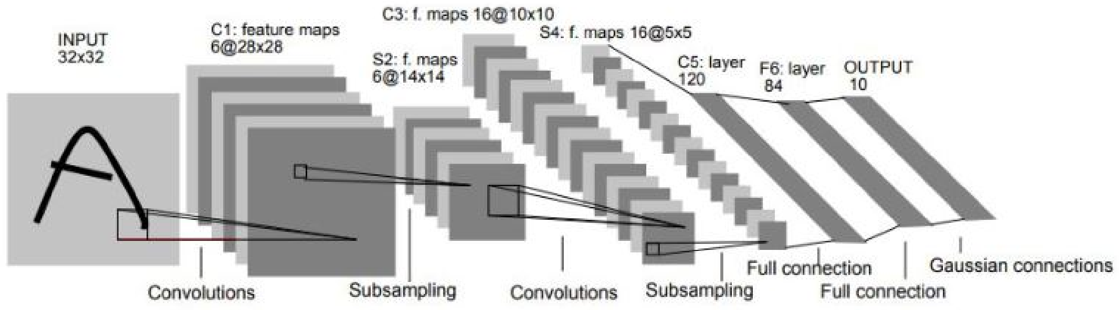

Figure 5. The architecture of the LeNet network that was used for MNIST hand written classification [4]

2.1.4 VGG architecture

VGG is a very deep CNN architecutre [5]. The input to the VGG network is a fixed size 224×224 RGB images.

The image is passed through a set of convolutional layers. The used filters have very small receptive field of size 3×3 (which is the smallest size to capture the first order neighborhood of the center). In one of the configurations, it also utilizes 1×1 convolution filters, which can be seen as a linear transformation of the input channels (followed by non-linearity). For all the convolution layers, the convolution strides

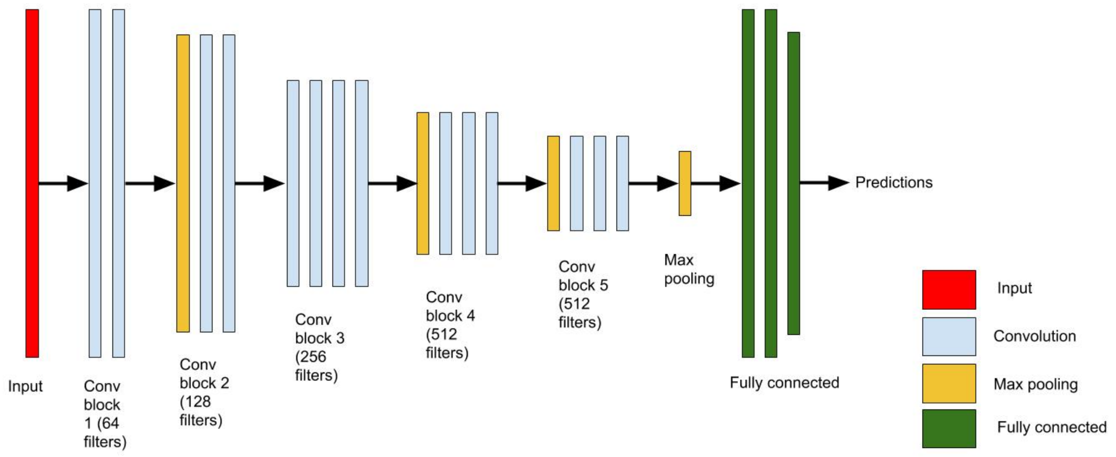
Input
Conv block 1 (64 filters)
Conv block 2 (128 filters)
Conv block 3 (256 filters)
Conv block 4 (512 filters)
Conv block 5 (512 filters)
Max pooling
Fully connected
Predictions
Input
Convolution
Max pooling
Fully connected

Figure 6. The architecture of the VGG16 network [5]

are fixed to 1 pixel. Since the convolution filters are of size 3×3, padding of one pixel is added to preserve the spatial resolution. There are 5 max pooling layers following some of the convolution layers. The max pooling is a 2×2 pooling with stride of 2 pixels. For all hidden layers, ReLU non-linearity is used. There are several configurations of VGG network. The last 3 layers before the softmax classifier are 3 dense layers. The first 2 dense layers have 4096 channels, while the third has 1000 channels.

There are several configurations of the VGG network. The most commonly used is the VGG16 (Figure 6). The different configurations are illustrated in Table 2 [5].

The VGG network configurations achieved very good results when tested on the ImageNet data. Table 3 shows the error rates achieved by each of the configurations shown in Figure 2. The major drawback of using VGG is the very slow training time due to the very deep architecture used.

2.1.5 ResNet architecture

Conceptually, going deeper in CNN is supposed to help achieving better results and allow better function representation between the input and the output [1]. However, in practice, going very deep with CNNs may lead to the problem of gradient vanishing [57]. As the gradient is back-propagated to earlier layers, repeated multiplication may make the gradient tend to zero. As a result, as the network goes deeper, its performance gets saturated or even starts degrading rapidly. There have been several ways suggested to overcome the problem of gradient vanishing. For example, in [6] auxiliary loss in a middle layer as extra supervision was added. These ideas did not address the problem properly.

Table 2. different configurations of VGG architecture

[5]

VGG Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weigh layers	16 weigh layers	19 weigh layers
Input image (224*224 images)					
Conv3 -64	Conv3-64 LRN	Conv3-64 Conv3-64	Conv3-64 Conv3-64	Conv3-64 Conv3-64	Conv3-64 Conv3-64
Maxpooling					
Conv3 -128	Conv3 -128	Conv3 -128 Conv3 -128	Conv3 -128 Conv3 -128	Conv3 -128 Conv3 -128	Conv3 -128 Conv3 -128
Maxpooling					
Conv3 -256 Conv3 -256	Conv3 -256 Conv3 -256	Conv3 -256 Conv3 -256	Conv3 -256 Conv3 -256 Conv1 -256	Conv3 -256 Conv3 -256 Conv3 -256	Conv3 -256 Conv3 -256 Conv3 -256 Conv3 -256
Maxpooling					
Conv3 -512 Conv3 -512	Conv3 -512 Conv3 -512	Conv3 -512 Conv3 -512	Conv3 -512 Conv3 -512 Conv3 -512	Conv3 -512 Conv3 -512 Conv3 -512	Conv3 -512 Conv3 -512 Conv3 -512 Conv3 -512
Maxpooling					
Conv3 -512 Conv3 -512	Conv3 -512 Conv3 -512	Conv3 -512 Conv3 -512	Conv3 -512 Conv3 -512 Conv1 -512	Conv3 -512 Conv3 -512 Conv3 -512	Conv3 -512 Conv3 -512 Conv3 -512 Conv3 -512
Maxpooling					
Conv3 -512 Conv3 -512	Conv3 -512 Conv3 -512	Conv3 -512 Conv3 -512	Conv3 -512 Conv3 -512 Conv1 -512	Conv3 -512 Conv3 -512 Conv3 -512	Conv3 -512 Conv3 -512 Conv3 -512 Conv3 -512
Maxpooling					
Fully connected 4096					
Fully connected 4096					
Fully connected 1000					
Softmax					

Table 3. The error rates achieved by each of the VGG network configurations when tested on ImageNet dataset

Configuration	**Error Rate**
VGG-11	10.4%
VGG-11 (LRN)	10.5%
VGG-13	9.9%
VGG-16 (Conv1)	9.4%
VGG-16	8.8%
VGG-19	9.0%

Table 4. The different configurations of ResNet architecture [1]

Layer Name	Outputsize	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112*112	7*7,64, stride 2				
conv2_x	56*56	3*3 max pool, stride 2				
		$\begin{bmatrix}3*3*,64\\3*3*,64\end{bmatrix}*2$	$\begin{bmatrix}3*3*,64\\3*3*,64\end{bmatrix}*3$	$\begin{bmatrix}1*1*,64\\3*3*,64\\1*1,256\end{bmatrix}*3$	$\begin{bmatrix}1*1*,64\\3*3*,64\\1*1,256\end{bmatrix}*3$	$\begin{bmatrix}1*1*,64\\3*3*,64\\1*1,256\end{bmatrix}*3$
conv3_x	28*28	$\begin{bmatrix}3*3*,128\\3*3*,128\end{bmatrix}*2$	$\begin{bmatrix}3*3*,128\\3*3*,128\end{bmatrix}*4$	$\begin{bmatrix}1*1*,128\\3*3*,128\\1*1,512\end{bmatrix}*4$	$\begin{bmatrix}1*1*,128\\3*3*,128\\1*1,512\end{bmatrix}*4$	$\begin{bmatrix}1*1*,128\\3*3*,128\\1*1,512\end{bmatrix}*8$
conv4_x	14*14	$\begin{bmatrix}3*3*,256\\3*3*,256\end{bmatrix}*2$	$\begin{bmatrix}3*3*,256\\3*3*,256\end{bmatrix}*6$	$\begin{bmatrix}1*1*,256\\3*3*,256\\1*1,1024\end{bmatrix}*6$	$\begin{bmatrix}1*1*,256\\3*3*,256\\1*1,1024\end{bmatrix}*23$	$\begin{bmatrix}1*1*,256\\3*3*,256\\1*1,1024\end{bmatrix}*36$
conv5_x	7*7	$\begin{bmatrix}3*3*,512\\3*3*,512\end{bmatrix}*2$	$\begin{bmatrix}3*3*,512\\3*3*,512\end{bmatrix}*2$	$\begin{bmatrix}1*1*,512\\3*3*,512\\1*1,2048\end{bmatrix}*3$	$\begin{bmatrix}1*1*,512\\3*3*,512\\1*1,2048\end{bmatrix}*3$	$\begin{bmatrix}1*1*,512\\3*3*,512\\1*1,2048\end{bmatrix}*3$
	1*1	average pooling, 1000 fully connected, softmax				

The ResNet proposed in [1], suggests a solution for this problem that allows for going deeper with convolution. The basic idea of ResNet is to introduce the concept of identity shortcut connections. This type of connection skips one or more layers, and transfers the input to the output directly. Figure 7 [1] illustrates the residual blocks with the skip connection. Also Figure 8 shows an example of a full ResNEt network using the residual block. Using this skip connections, more layers could be stacked without hurting the network performance, because simply the network can still learn the identity function. This ensures that the blocks with residual connections will not perform worse the shallower counterparts.

As in the VGG, there have been multiple configurations of the ResNet. Table 4 shows the various configurations used with the ResNet. Also Table 5 shows the obtained results with the different ResNet configurations on the ImageNEt dataset. It can be observed that going deeper with ResNet enhanced the results and achieved lower error rates.

2.1.6 Inception architecture

The main motivation behind the inception architecture is the wide variability of the objects sizes and locations in the images. This makes capturing the object with convolution difficult when using a single size filter per layer [6]. Figure 9 shows an example from the WBC dataset, where the WBCs vary in the size significantly.

In order to address this problem, the inception architecture was proposed. The main idea in this architecture, is to use multiple filters of different sizes in the same

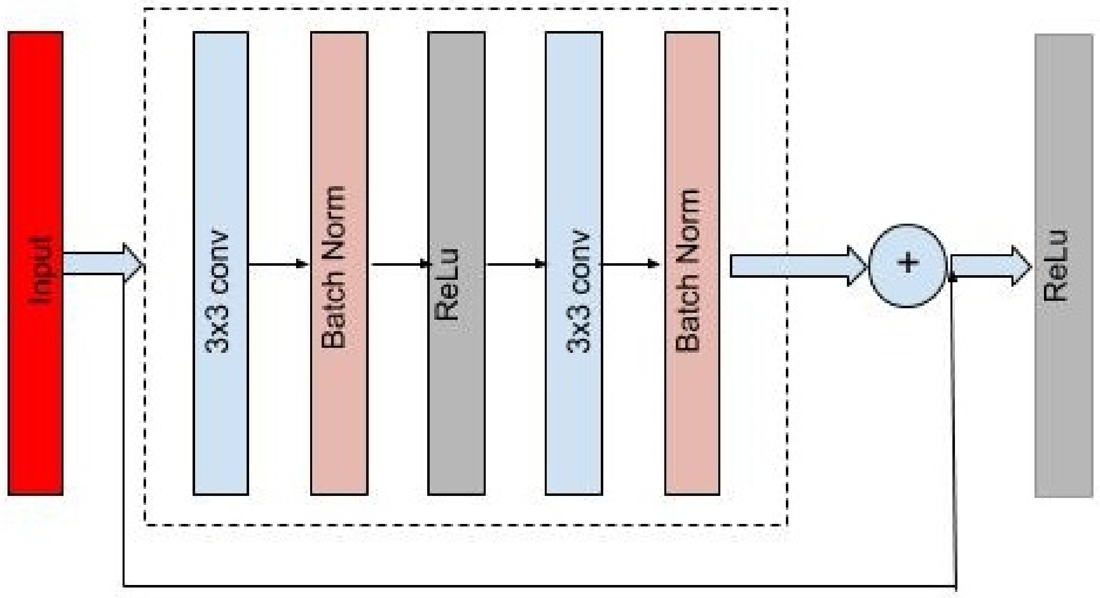

Figure 7. The residual block used in the ResNet architecture. The input could be passed directly passed to the output. [1]

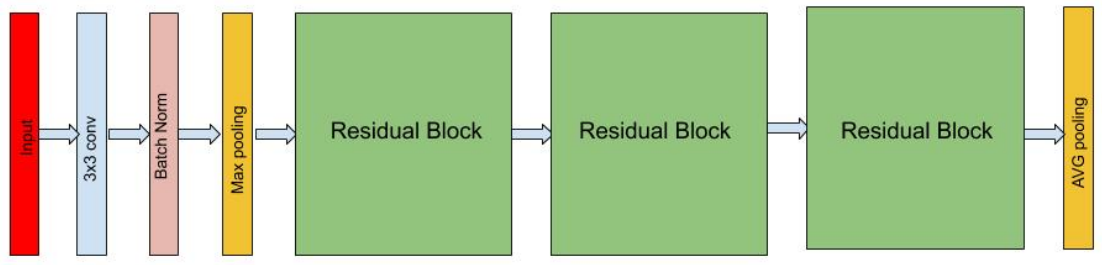

Figure 8. An example of a ResNet network with 3 residual blocks. This network could be expanded by adding more residual blocks.[1]

Table 5. The error rates on ImageNet dataset obtained using a 34 layers without skip connections, 34, 50, 101 and 152 layers with skip connections.

Configuration	**Error Rate**
plain-34	10.2%
ResNet-34	7.76%
ResNet-50	6.71%
ResNet-101	6.05%
ResNet-152	5.71%

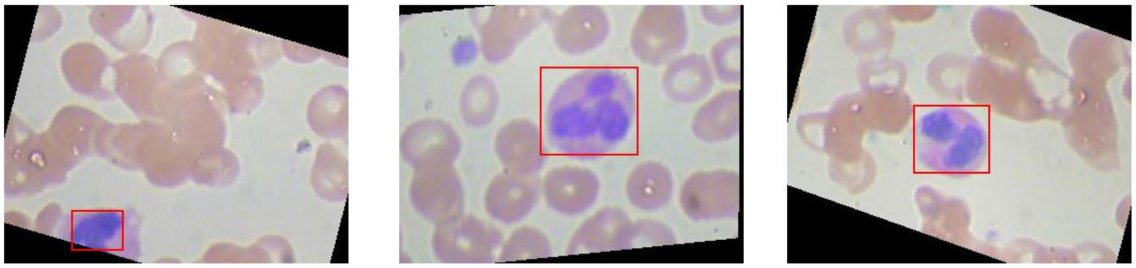

Figure 9. Three examples of WBCs images where the size of WBCs varies significantly. This makes it harder to use a single a convolutional filter to capture the image features.

layer. In order to limit the number of computations and reduce the number of channels, the authors suggested using the 1×1 convolution. This type of convolution acts as an averaging filter over channels depth, which reduces the number of the channels as we go deeper in the network [6]. Figure 10 shows an example of how the multiple filter size are used at the same layer and then concatenated to have more flexible feature maps that can capture objects at different scales and how classification blocks are used at intermediate layers.

Having a relatively very deep architecture for inception network, one of the possible limitations was the ability to back propagate the gradients through the whole depth. Since each layer in the network is supposed to generate some discriminative features, adding auxiliary classifiers was one of solution to overcome the gradient vanishing problem. Adding these auxiliary classifiers aims mainly to enhance the discriminative power of the intermediate layers. These classifiers also increase the gradient signal that gets propagated back, and provide additional regularization. Figure 11 shows the entire architecture of the inception network with two auxiliary classifiers. When tested on the ImageNet dataset, the error rate was 6.67%.

2.2 Customized deep learning models

In addition to using the pretrained deep learning models, two customized deep learning models are tested. These models are not pretrained.

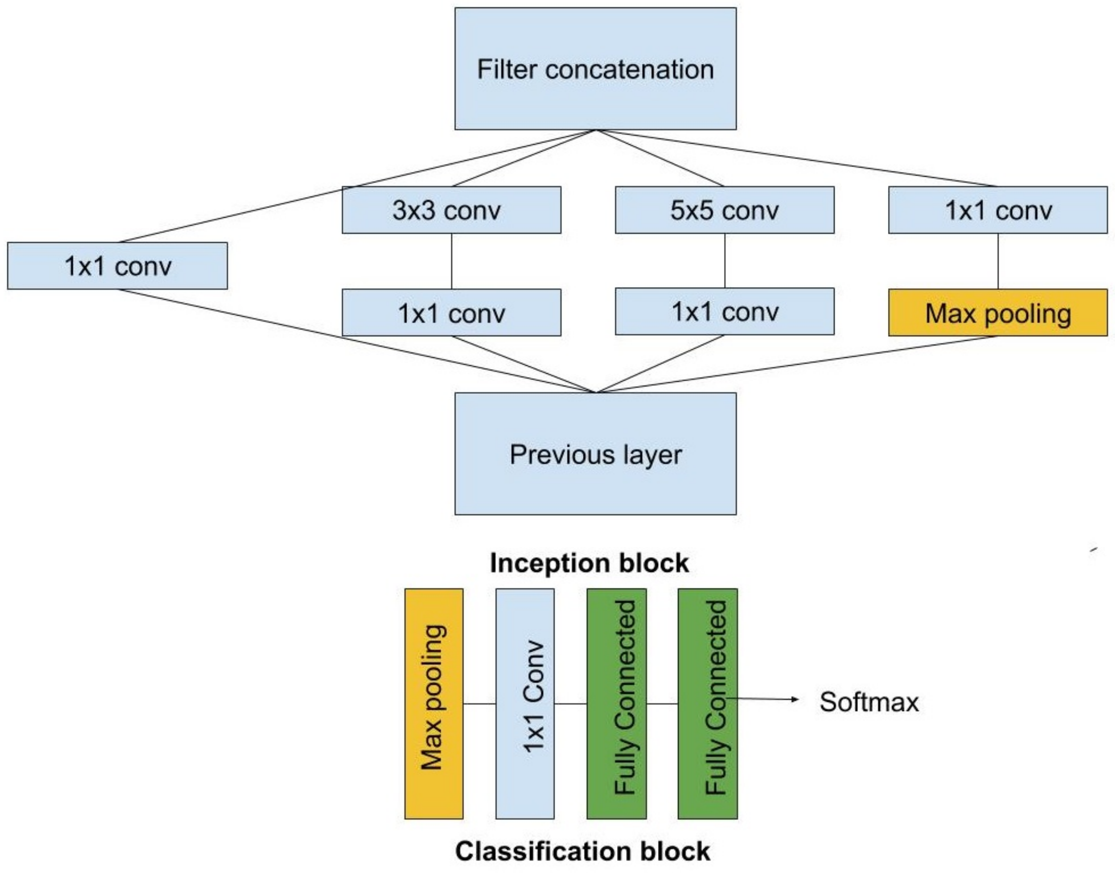

Figure 10. The two main building blocks: the inception block that concatenates feature maps obtained from filters of different sizes and the classification block at intermediate layers

2.2.1 Customized model 1

The first customized model is a classical CNN. This CNN could be divided into 4 main blocks:

1. Block 1: This block contains a convolution layer with filter size 3×3 and it generates 32 feature maps. The second layer in the block is a max pooling layer of size 2×2 and stride of one. The third layer is a dropout layer of proportion 0.2. The main reason to add the dropout layer is to prevent the overfitting by randomly dropping out some nodes.

2. Block 2: This block contains a convolution layer with filter size 3×3 and it generates 64 feature maps. The second layer in the block is a max pooling layer of size 2×2 and stride of one. The third layer is a dropout layer of proportion 0.2.

3. Block 3: This block contains a convolution layer with filter size 3×3 and it

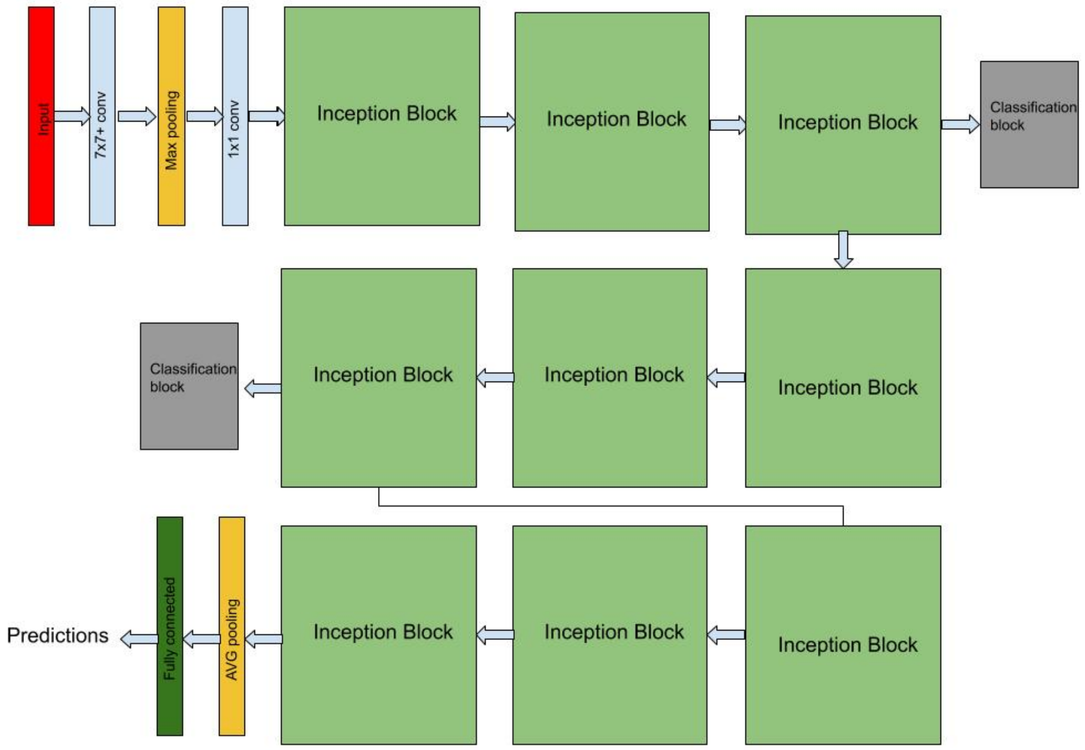

Figure 11. The entire inception network architecture with two auxiliary classifier. The orange box is the stem, which has some preliminary convolutions. The purple boxes are auxiliary classifiers. The wide parts are the inception modules. [6]

generates 128 feature maps. The second layer in the block is a max pooling layer of size 2×2 and stride of one. The third layer is a dropout layer of proportion 0.2.

4. Block 4: In this block, a flatten layer is added, followed by 3 dense layers of sizes 64, 128 and 64 respectively. The used activation for these layers is ReLU. The last layer is a soft max classification layer with 4 output nodes.

Figure 12 illustrates the architecture of the used customized CNN model.

2.2.2 Customized model 2

The second customized model is a deeper model. The main building block in this model is composed of 6 layers:

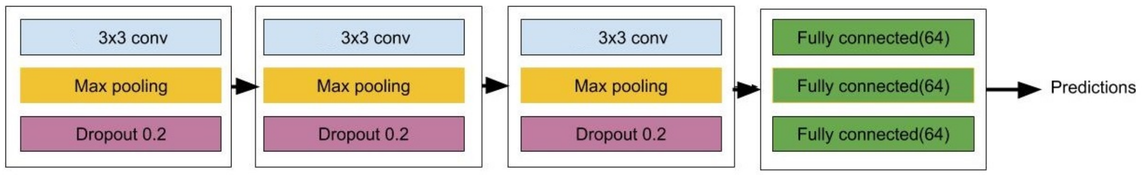

Figure 12. The architecture of the customized used CNN architecture. It contains 4 blocks. The first 3 block are convolution, max pooling and dropout, while the fourth contains 3 dense layer of sizes 64, 128 and 64 respectively.

1. Layer 1: It contains a number n feature maps with filter size 1×1. These 1×1 convolutions are used for channel wise pooling and it is useful in dimensionality reduction.
2. Layer 2: Batch normalization layer: this layer allows for using higher learning rates and be less careful about initialization. This allows for faster conversion times during the training phase. It also acts as a regularizer in some cases due its effect in reducing overfitting
3. Layer 3: It contains a number n feature maps with filter size 1×1.
4. Layer 4: Batch normalization layer
5. Layer 5: It contains a number n feature maps with filter size 3×3.
6. Layer 6: Batch normalization layer

The convolution mode used in the main building block is the 'same' mode, which maintains the same input input shape. The output of this building block is the concatenation of layer 4 and layer 6 feature maps.

For the entire model architecture, 3 introductory layers are applied where the input is passed first to a batch normalization layer, followed by convolution layer with 12 filters and kernel size $=5 \times 5$, then another batch normalization. After these introductory layers, the building block followed by max pooling of size 2×2 are applied for 5 times with $n = 12, 24, 32, 24, 18$ respectively followed by the building block of $n = 12$. After completing these convolution and pooling layers, a dense layer with sigmoid activation function. Figure 13 shows the building block and the entire model architecture.

2.3 Region of interest extraction

In this section, two methods are discussed to extract the Region of Interest (ROI), the first method is color based method, and the second method is a deep learning based method.

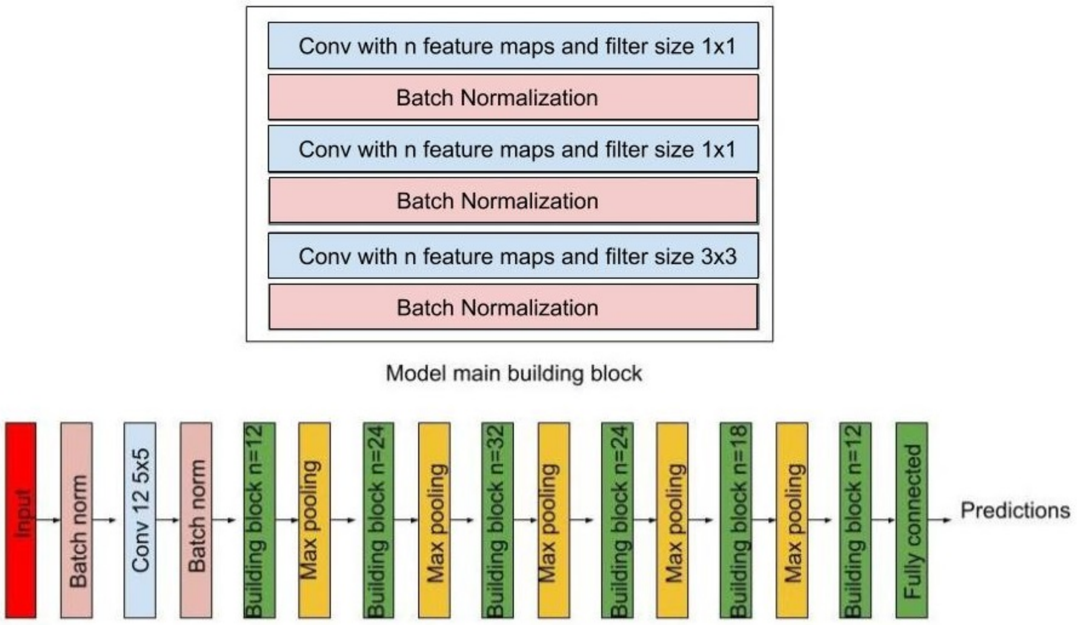

Figure 13. The architecture of the customized used CNN architecture. The main building block is illustrated in the upper row, and the entire model architecture in the lower row

2.3.1 Color based ROI extraction

Based on the color pattern of the WBCs in the dataset, it is obvious that the WBCs all fall in the same color range. The proposed algorithm to extract an ROI around the WBCs contains 4 main steps:

1. Blur the image for noise reduction using Gaussian filter of size 3×23.
2. Transform the image to HSV color space
3. Binarize the image within the color range containing all WBCs; typically in the range of RGB colors (80, 60, 140) to (255, 255, 255)
4. Find the largest contour in the binary image using contour tracing algorithm.

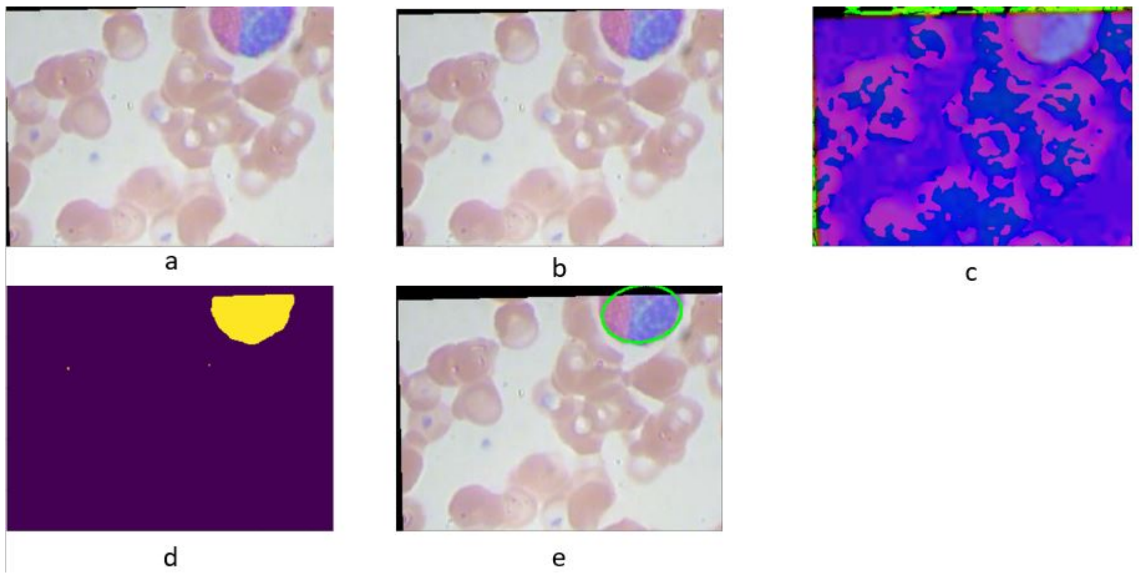

Figure 14. The steps of the color based ROI: (a) the original image, (b) the blurred image, (c) the image in HSV space, (d) the binarized image based on the color range, and (e) the extracted contour form the contour tracing algorithm

```
Algorithm 1: Contour tracing
  Result: Return list of contour pixels
  1- Start scanning the image from the top left corner until a pixel of value 1 is
     found. Call this object pixel $b_0$, and its west background (value 0)
     neighbour $c_0$;
  2 - Check the first order neighborhood of $b_0$, starting at $c_0$ and continue in
     clockwise direction. Let $b_1$ be the first pixel of value 1 encountered, and $c_1$
     be the pixel of value 0 encountered preceding $b_1$. Save the locations of $b_0$
     and $b_1$, to the contour pixels list ;
  3 - Let $b = b_1$ and $c = c_1$;
  4- Let the first order neighborhood of $b$, starting at $c$ and continuing in a
     clockwise direction, be denoted as $n_1, n_2, ..., n_8$. Find the first object
     neighbour $n_k$ in this sequence. ;
  5- Let $b = n_k$, $c = n_{k-1}$. Add $b$ to the contour pixels list.;
  while while $b \neq b_0$ do
      6- Repeat step 4 and 5
  end
```

In order to find the largest contour in a binary image, contour tracing algorithm,illustrated in Algorithm 1 is used [58] . This algorithm is based on identifying the external borders and the hole borders. Figure 3.2.1 shows the original image, the blurred image, the color thresholded image in HSV space, and the extracted contour.

2.3.2 Mask R-CNN ROI extraction

Mask R-CNN is a deep learning approach used for extracting the ROI of the objects in the image, in addition it also defines the object mask which could be used in instance segmentation [59]. The Mask R-CNN is based on the Region CNN (R-CNN) [60] and the faster R-CNN [61].

In R-CNN, a pre-trained CNN is used. The last layer in the model is then retrained with the required number of classes to detect. The third step is to extract the ROIs in each image. All ROIs are then reshaped to match the original image size. The ROIs are then to a binary SVM classifiers, one classifier per class. Finally, a linear regression model is used to generate tighter bounding boxes for each identified object in the image.

The main limitation of the R-CNN is its high computational cost and slow performance. The reason behind this performance issue is the need to input each ROI to the CNN which is inefficient. Typically it takes up to 50 seconds to make predictions on single image [60].

In order to overcome this performance problem, fast R-CNN was proposed in [62]. The intuition behind Fast R-CNN is to input the entire image to the CNN and then find a way all the ROIs can share the computations instead of feeding each ROI to the CNN. To achieve this Fast R-CNN perform 3 main steps:

1. Feed the input image to a pretrained CNN to generate feature maps
2. From the extracted feature maps, ROIs are extracted and fed to ROI pooling layer to reshape all ROIs to the same size.
3. Softmax layers is then used after a fully connected layer to get the output classes, in addition, linear regression is applied in parallel to get bounding box coordinates of each of the predicted objects.

The bottleneck in the Fast R-CNN is the selective search to find the ROIs within the feature maps. This is still computationally process. To overcome this problem Faster R-CNN was proposed [61]. In the Faster R-CNN, the selective search for ROI is replaced with Region Proposal Network "RPN". The RPN takes the feature maps an inputs and generates set of object proposals, each is associated with a score that expresses the RPN confidence. As in the FAST R-CNN, the proposals are then fed to the ROI pooling layer, followed by the softmax and the linear regression layer. Figure 15 shows architecture of the R-CNN [63].

For the Mask R-CNN, it is built on the top of Faster R-CNN. It allows the model to do both object detection and pixel- wise instance segmentation as well. The main intuition behind the Mask R-CNN is to add a branch to the Faster R-CNN that generates a binary mask which tells whether a pixel is a part of a certain object or not. To enhance the generated mask sensitivity and make it precise enough to do the instance segmentation task, The Mask R-CNN uses an algorithm called ROI align. This algorithm uses bilinear interpolation to determine the precise location of the pixel in the original image using its location in the feature map. Figure 16 shows the building blocks of the Mask R-CNN architecture.

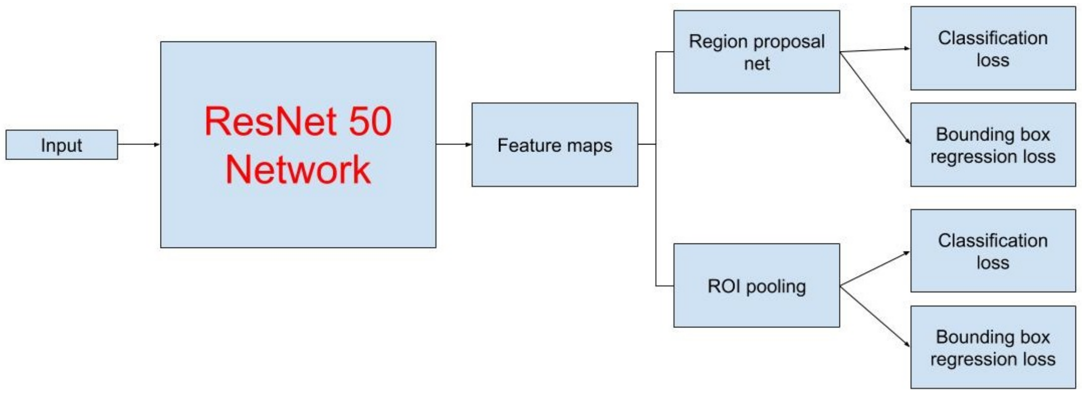

Figure 15. The Faster R-CNN architecture, where ResNet-50 CNN is used to generate feature maps. The feature maps are then fed to RPN layer to generate region proposals, then to ROI pooling to resize these proposal, finally, softmax and linear regression are used for object detection and localization respectively

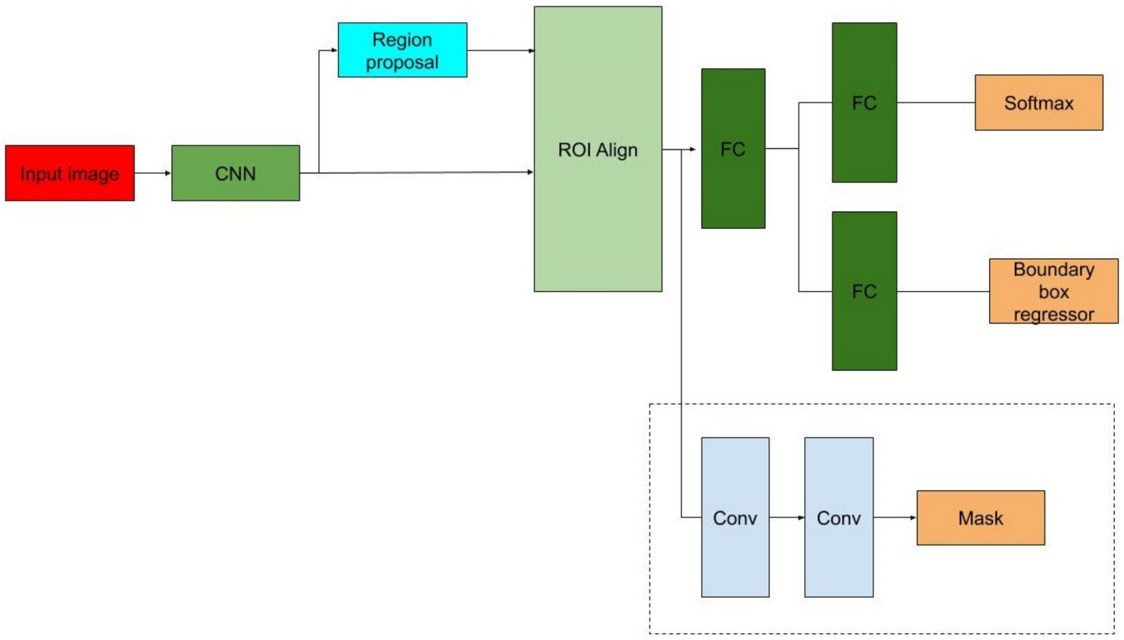

Figure 16. The Mask R-CNN architecture. It shares the same architecture with Faster R-CNN with an extra branch for binary mask generation. To map the mask accurately to the original image, ROI align algorithm is used.

CHAPTER 3

EXPERIMENTAL RESULTS AND DISCUSSION

3.1 Dataset description

The dataset used in this study consists of 12,444 microscopic images for blood cells. The data contains images for 4 classes of WBCs: (i) eosinophils, (ii) eosinophils, (iii) monocytes, and (iv) neutrophil. The dataset is divided into training (75%) and testing (25%). The distribution of the 4 classes in both training and testing datasets is showm in Table 6. All the images are of a depth of 24 bits and a resolution of 320×240 pixels. Figure 1 shows samples of the 4 classes of WBCs.

In addition to the classification dataset that comes with ground truth for the classification, another part of the dataset comes with ground truth of the bounding box coordinates of each WBC in the image. This data is used for ROI extraction task. The ROI dataset contains 364 images with bounding box annotation coordinates given by experts for red and white blood cells which is used as ground truth for evaluation. Figure 17 shows samples of the bounding boxes given by experts. Both datasets are publicly available online at: `https://www.kaggle.com/paultimothymooney/blood-cells`

3.2 ROI extraction experiments

In this section, two experimental results are discussed for WBCs extraction. The first experiment is using a color based approch discussed in subsection 2.3.1, while

Table 6. The number of available images in each of WBC types (i) eosinophils, (ii) eosinophils, (iii) monocytes, and (iv) neutrophil for both training and testing

	Training	**Testing**
Eosinophils	2,497	623
Lymphocytes	2,483	620
Monocytes	2,478	620
Neutrophil	2,499	624

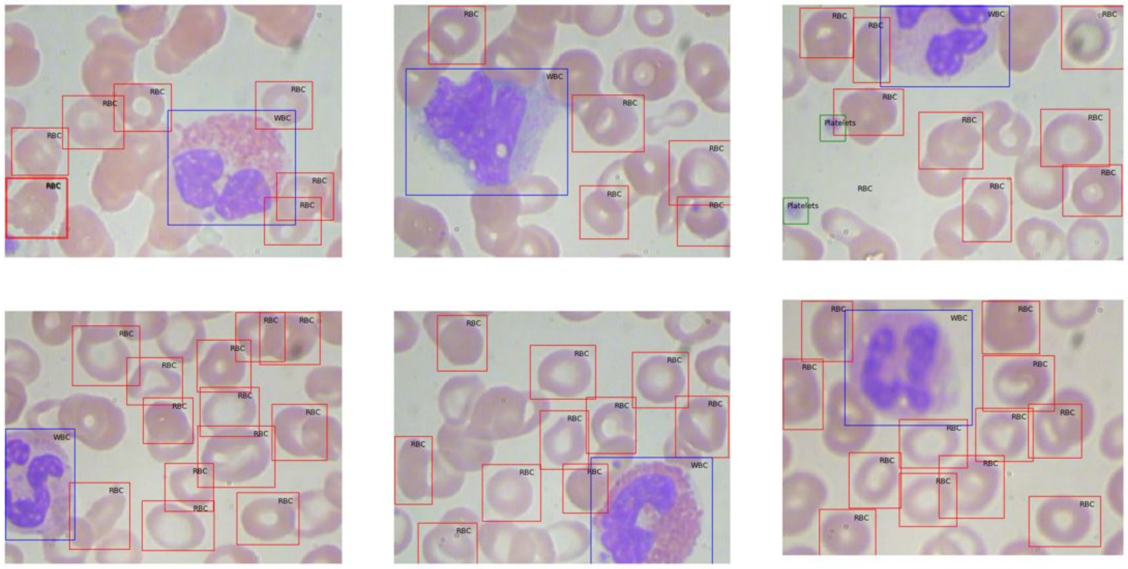

Figure 17. Samples of ground truth bounding box for red and white blood cells as given by experts.

the second experiment is using Mask R-CNN discussed in subsection 2.3.2.

3.2.1 Color Based ROI extraction results

Using the algorithm described in subsection 2.3.1, a contour defining surrounding the WBCs was generated. The minimum and maximum in both X and Y directions were used to define a bounding box surroundin the WBCs. To check how precise the bounding box encloses the WBC in the image, the Intersection over Union (IoU) is used between the ground truth bounding box and the output bounding box.

The average IoU obtained using this algorithm is 0.59. The summary statistics of the IoU is shown in Table 7. Also, Figure 18 shows the histogram of the IoU when tested on the testing dataset (144 images from the annotated images), and Figure 19 shows 6 samples of the bounding boxes obtained from the algorithm (black) and the ground truth bounding boxes (blue).

The obtained results and the bounding box visualization show that this algorithm is very sensitive to the color contrast and if there is background pixels falling in the same color range and are adjacent to the WBC, it will be considered a part of the WBC ROI. Also if any part of the WBC does not fall in the designated color range, this part will be excluded from the ROI.

3.2.2 Mask R-CNN ROI extraction results

In this experiment a deep learning based approach is used for ROI extraction to compare with the results obtained using the color based approach. The used algorithm is the Mask R-CNN described in 2.3.2, where two pretrained networks are used: (i)

VGG16 and (ii) ResNet50. For the hyperparameters used, the number of epochs is 50, the batch size = 32, the used optimized is Adam optimizer. Since this algorithm is a supervised algorithm, the annotated data was divided into training (220 images) and testing (144 images).

As in the previous experiment, IoU was used to evaluate the ROI extraction. The mean IoU achieved is 0.71 when using VGG16 and 0.69 when using ResNet50. The minimum, maximum, standard deviation, and median of the obtained IoUs are listed in Table 8. The distribution of IOU obtained on the testing images is shown on Figure 20. and examples of the extracted ROI (orange) and the ground truth bounding box (blue) are shown in Figure 21.

From the obtained results, the Mask R-CNN with the VGG16 pretrained network gives the better overall ROI extraction results. However, it is obvious that the VGG16

Table 7. The IoU mean, minimum, maximum, standard deviation, and median when using the color based approach for RoI extraction

	Color based ROI extraction
Mean IoU	0.59
Minimum IoU	0.25
Maximum IoU	0.91
Standard deviation of IoU	0.12
Median IoU	0.6

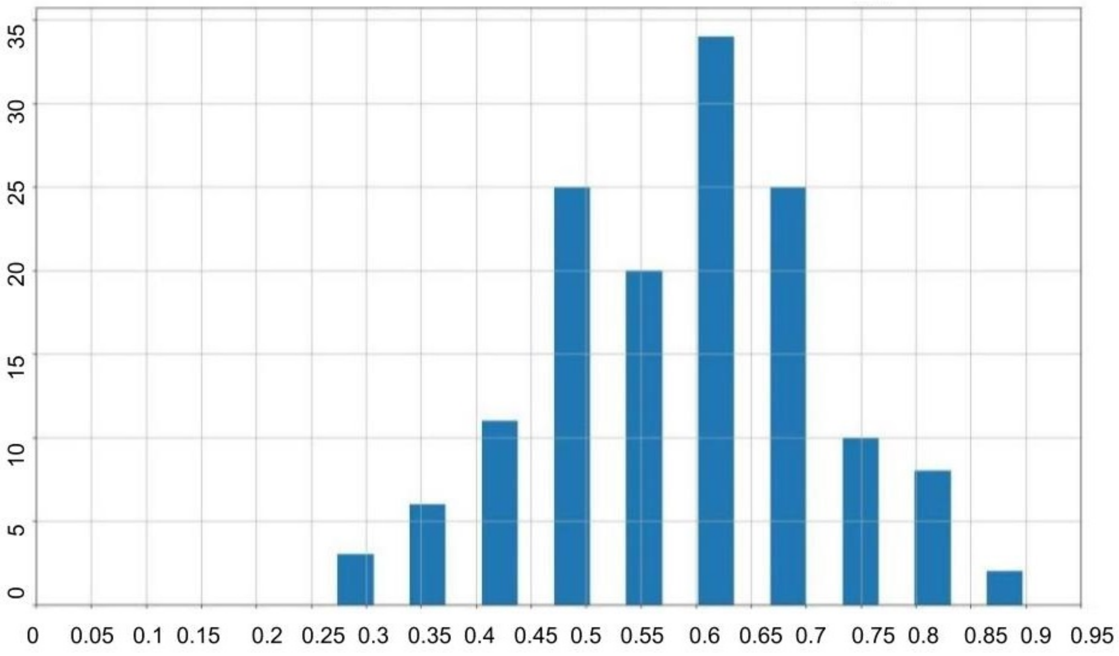

Figure 18. The histogram of the IoU when tested on the testing dataset (144 images from the annotated images) using color based ROI extraction.

and ResNet50 results are very close to each other, and they are both outperforming the color based method. This results show that including higher level features extracted by the Mask R-CNN is more powerful than utilizing just the color features.

3.3 Classification experiments using the whole images for training and testing

In this section, we proceed with the classification of the 4 different types of WBCs. Classification results will be discussed for all the models described in sections 2.1 and 2.2. For each of the CNN models, we will compare the classification results when using the whole images and when using the extracted ROIs. For each experiment, the accuracy and confusion matrix are reported.

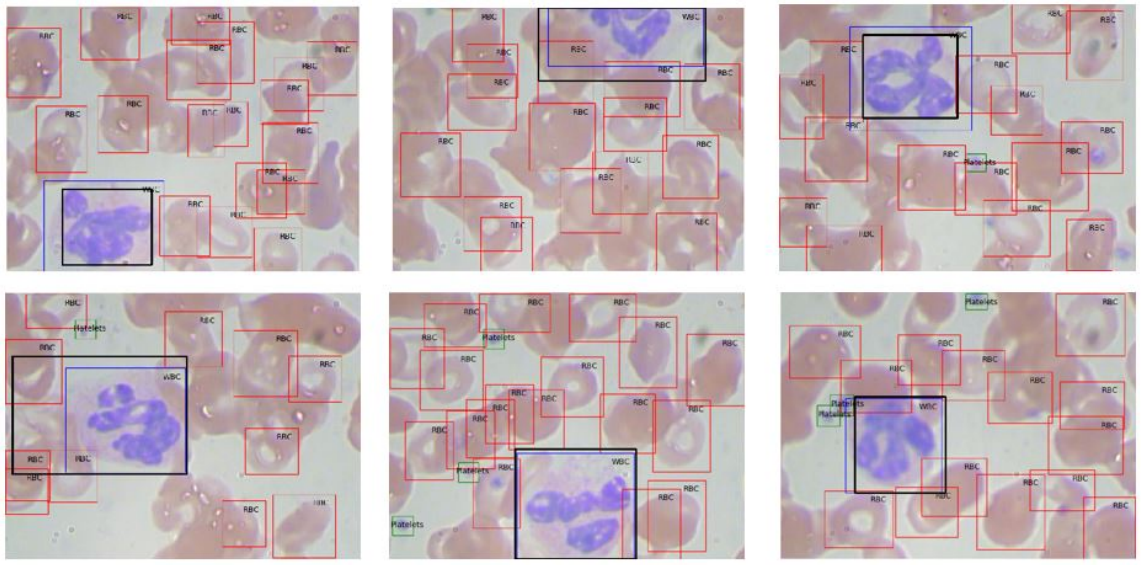

Figure 19. Examples of the bounding boxes obtained from the color based ROI extraction algorithm (black) and the ground truth bounding boxes (blue)

Table 8. The IoU mean, minimum, maximum, standard deviation, and median when using the mask R-CNN for RoI extraction for both VGG16 and ResNet50

	Mask R-CNN	
	VGG16	**ResNet50**
Mean IoU	0.71	0.69
Minimum IoU	0.06	0.002
Maximum IoU	0.92	0.93
Standard deviation of IoU	0.11	0.11
Median IoU	0.73	0.7

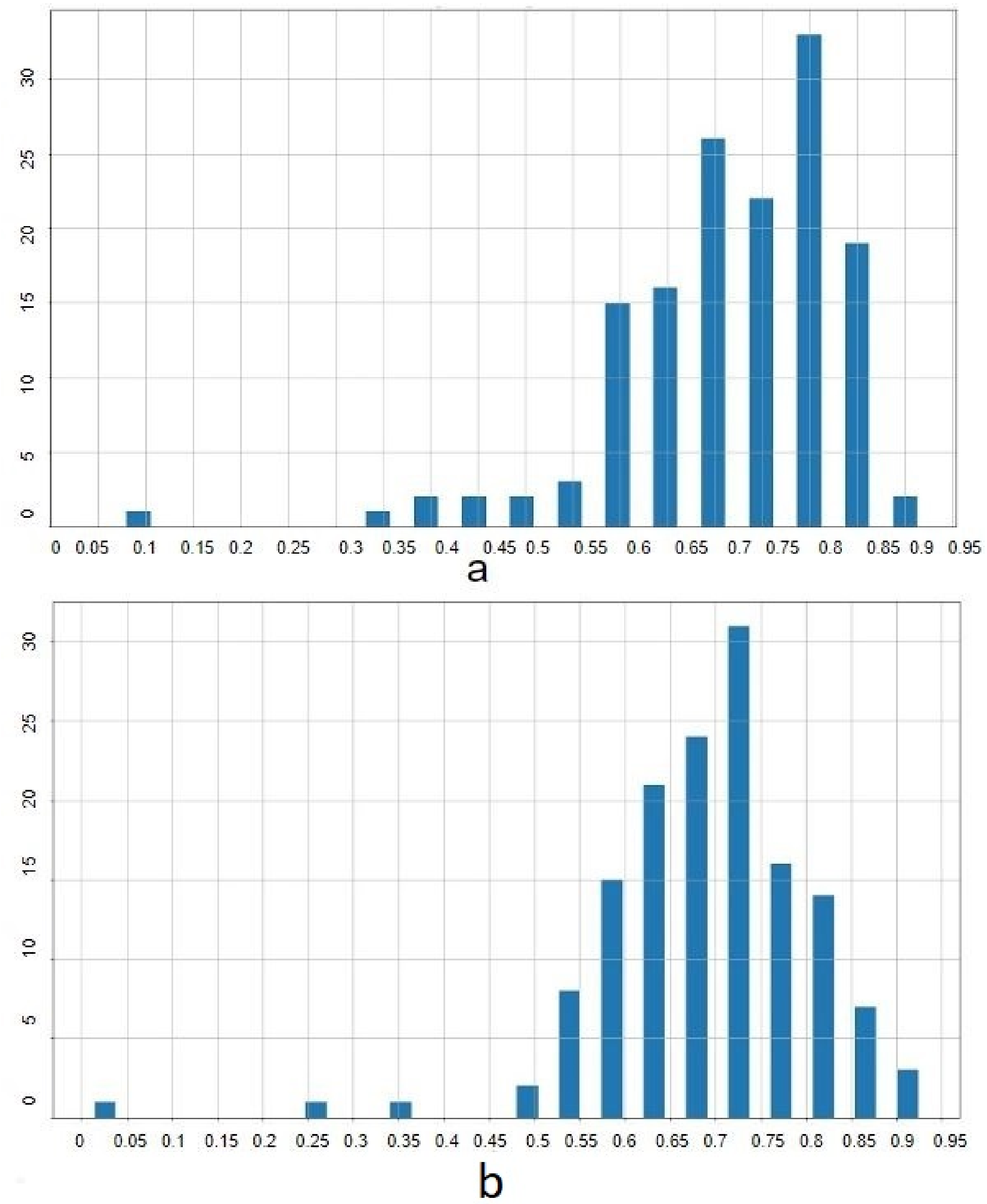

Figure 20. The histogram of the IoU when tested on the testing dataset (144 images from the annotated images) using Mask R-CNN with pretrained networks(a) VGG16 and (b) ResNet50

3.3.1 Classification using LeNet with the whole images

The LeNet was designed to be trained and tested on 32×32 gray scale images, this explains the low accuracy that was obtained. The achieved accuracy is 24.9%. This accuracy is not better than random guessing as the data contains 4 classes with approximately 25% per class. As a preprocessing step, another experiment was done using LeNet, where the images were converted to gray scale and resized to be 32×32. This experiment did not help improving the results. The accuracy is still 24.9%. Table 9 shows the confusion matrix obtained when using LeNet. It is obvious that all the testing images where classified as Neutrophil, this means that the classifier was not able to capture any pattern from the data.

3.3.2 Classification using VGG with the whole images

Since the simple LeNet architecture was not able to learn any pattern in the data, a more complex architecture was needed. In this experiment, the VGG16 architecture

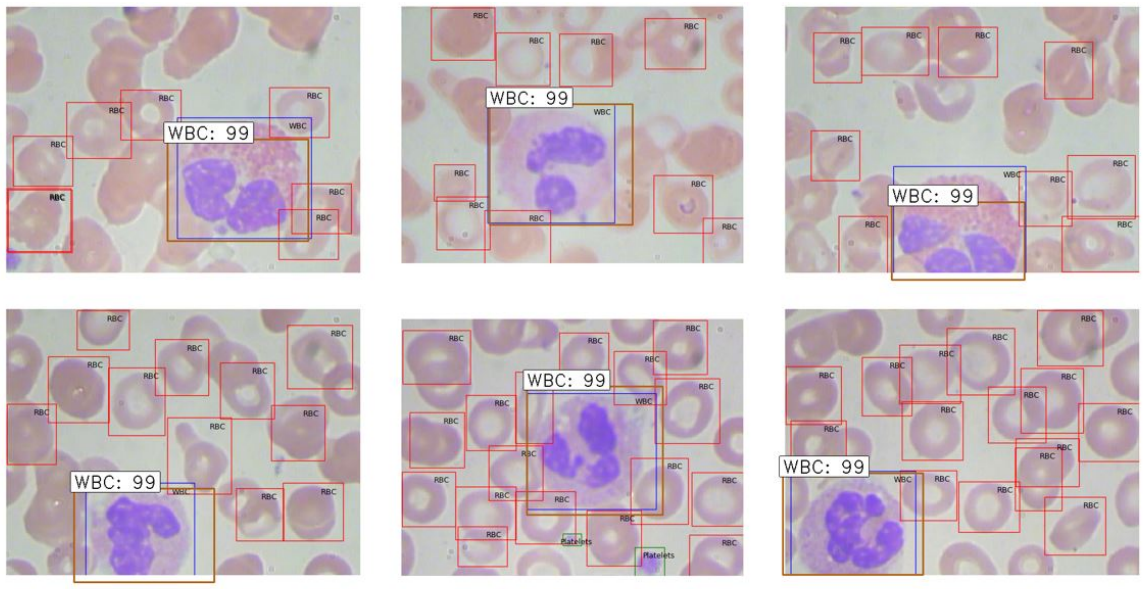

Figure 21. Examples of the bounding boxes obtained from the Mask R-CNN ROI extraction algorithm (Orange) and the ground truth bounding boxes (blue)

Table 9. The confusion matrix obtained when using LeNet. All the testing images where predicted to belong to Neutrophil class.

		True Label			
		EOSINOPHIL	LYMPHOCYTE	MONOCYTE	NEUTROPHIL
Predicted Label	EOSINOPHIL	0	0	0	0
	LYMPHOCYTE	0	0	0	0
	MONOCYTE	0	0	0	0
	NEUTROPHIL	623	620	620	624

discussed in 2.1.4 is used. The used VGG16 is pretrained using the ImageNet data, the only layer removed is the fully connected layer. It is replaced with a fully connected layer with 4 output nodes. The only layers trained in the model are the fully connected layer and the softmax layer. All the VGG convolution layers are not retrained again. The weights trained for ImageNet are used.

After using cross validation to obtain the optimal set of hyperparameters, the selected optimizer is Adadelta [64], the batch size is 32, learning rate = 1, and decay factor = 0.95.

Using these hyperparameters, and using 20 epochs for training, the accuracy obtained is 48.4%. The confusion matrix obtained by using VGG16 is shown in Table 10. Also, The training and testing accuracy over the 20 epochs is shown in Figure 22. The training accuracy kept increasing at each epoch, while the testing accuracy started to saturate at earlier stage. The VGG16 improved the accuracy compared to LeNet, but there is a big room for improvement that might be obtained using other models.

3.3.3 Classification using ResNet with the whole images

In this experiment, ResNet50 architecture described in 2.1.5 is used for classification with the whole images as input in the training and testing. The main advantage of ResNet50 is allowing for using very deep architecture with no worries about the gradient vanishing. The used ResNet50 is pretrained using the ImageNet data, the only layer removed is the fully connected layer. It is replaced with a fully connected layer with 4 output nodes. The only layers trained in the model are the fully connected layer and the softmax layer. All the ResNet50 convolution and batch normalization layers are not retrained again.The weights trained for ImageNet are used.

After using cross validation to obtain the optimal set of hyperparameters, the selected optimizer is Adam [64], the batch size is 32, learning rate = 0.001.

Using these hyperparameters, and using 20 epochs for training, the accuracy obtained is 74.6%. The confusion matrix obtained by using ResNet50 is shown in Table 11. Also, The training and testing accuracy over the 20 epochs is shown in Figure 23. The training accuracy kept increasing at each epoch, while the testing accuracy started to saturate at earlier stage. The ResNet50 improved the accuracy compared to VGG.

Table 10. The confusion matrix obtained when using VGG16.

		True Label			
		EOSINOPHIL	LYMPHOCYTE	MONOCYTE	NEUTROPHIL
Predicted Label	EOSINOPHIL	300	101	142	112
	LYMPHOCYTE	113	351	129	100
	MONOCYTE	162	81	246	54
	NEUTROPHIL	48	87	103	358

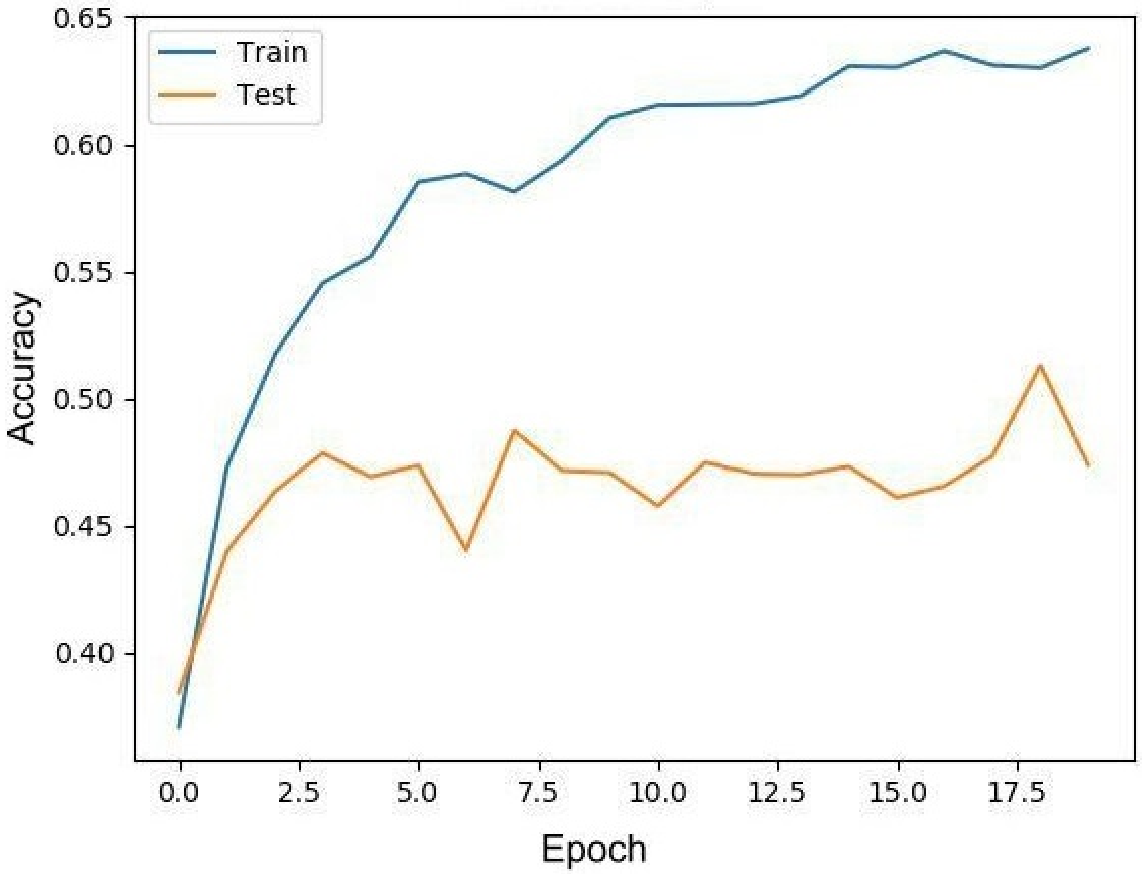

Figure 22. The training and testing history over the 20 epochs when using VGG16 model with the whole images.

3.3.4 Classification using Inception network with the whole images

In this experiment, the last pretrained network architecture, Inception architecture, described in 2.1.6 is used. The main advantage of this architecture is having multiple convolution filters with different sizes are used. This allows to capture features with different sizes at each layer. The used Inception network is pretrained using the ImageNet data, the only layer removed is the fully connected layer. It is replaced with a fully connected layer with 4 output nodes. The only layers trained in the model are the fully connected layer and the softmax layer. All the Inception network convolu-

Table 11. The confusion matrix obtained when using ResNet50.

		True Label			
		EOSINOPHIL	LYMPHOCYTE	MONOCYTE	NEUTROPHIL
Predicted Label	EOSINOPHIL	448	30	132	116
	LYMPHOCYTE	36	530	4	11
	MONOCYTE	12	29	388	6
	NEUTROPHIL	127	31	96	491

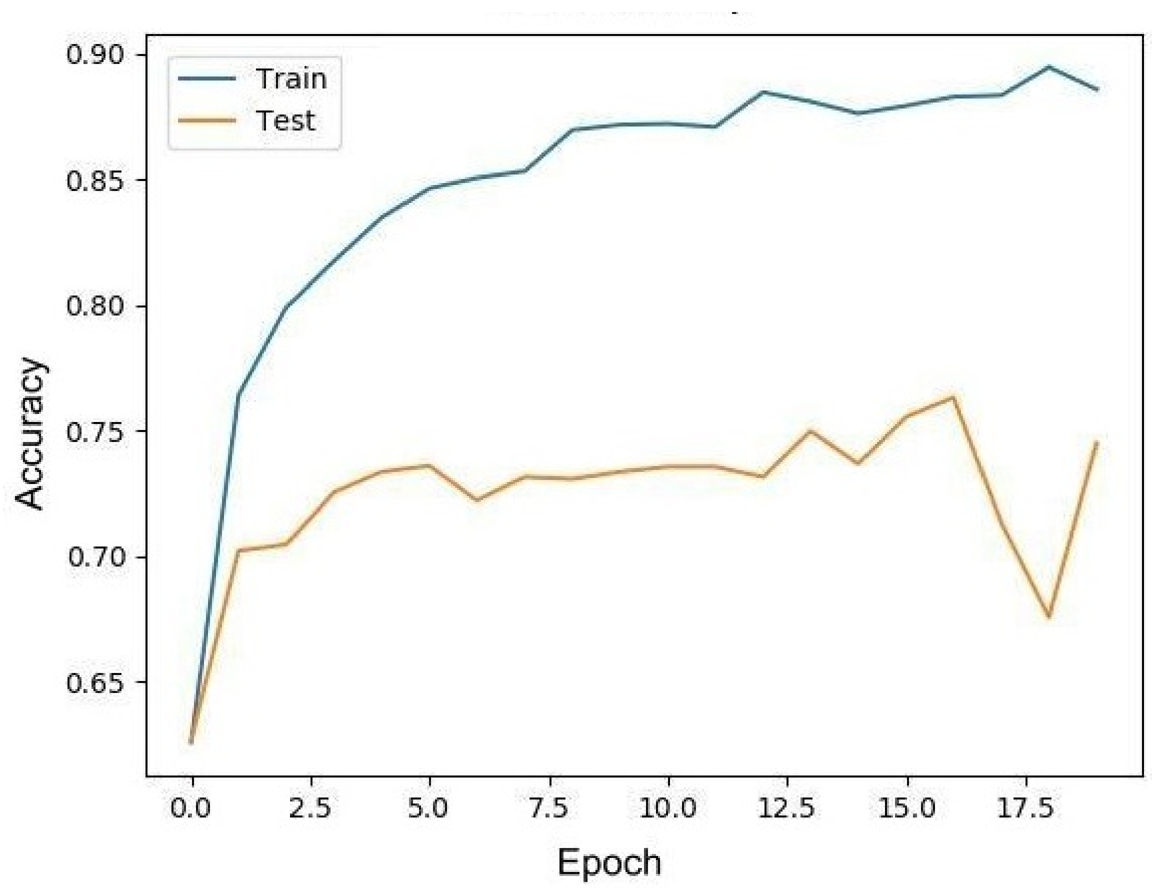

Figure 23. The training and testing history over the 20 epochs when using ResNet50 model with the whole images.

tion and batch normalization layers are not retrained again.The weights trained for ImageNet are used.

After using cross validation to obtain the optimal set of hyperparameters, the selected optimizer is ”Rmsprop” [64], the batch size is 32, learning rate = 0.001.

Using these hyperparameters, and using 20 epochs for training, the accuracy obtained is 43%. The confusion matrix obtained by using ResNet50 is shown in Table 12. Also, The training and testing accuracy over the 20 epochs is shown in Figure 24. The training accuracy kept increasing at each epoch, while the testing accuracy started to saturate at earlier stage.

The inception network results are worse than the ResNet50 and is close to the VGG results. What is inserting in the Inception results is having no images at all classified to belong to Neutrophil class.

The above experiments show that the best model performance obtained using the ResNet50 architecture. The next two subsections will discuss two customized models trained from scratch to classify the WBCs images. The main intuition behind these experiments is having the data not sharing too much shape features with the ImageNet dataset, thus, it might be better to train some models from scratch.

Table 12. The confusion matrix obtained when using Inception network.

		True Label			
		EOSINOPHIL	LYMPHOCYTE	MONOCYTE	NEUTROPHIL
Predicted label	EOSINOPHIL	392	159	258	370
	LYMPHOCYTE	183	405	69	148
	MONOCYTE	48	56	293	106
	NEUTROPHIL	0	0	0	0

3.3.5 Classification using the first customized model with the whole images

In this experiment, the first customized model discussed in 2.2.1 is used. This model is trained from scratch and it is relatively a simple model compared to the pretrained architectures. This simple architecture leads to less number of parameters to be learnt. This allows the learning to converge with limited number of training samples.

As preprocessing step, all the images were first resized to smaller size, 60×60, and normalized to be between 0 and 1 by dividing all pixel values by 255. Using

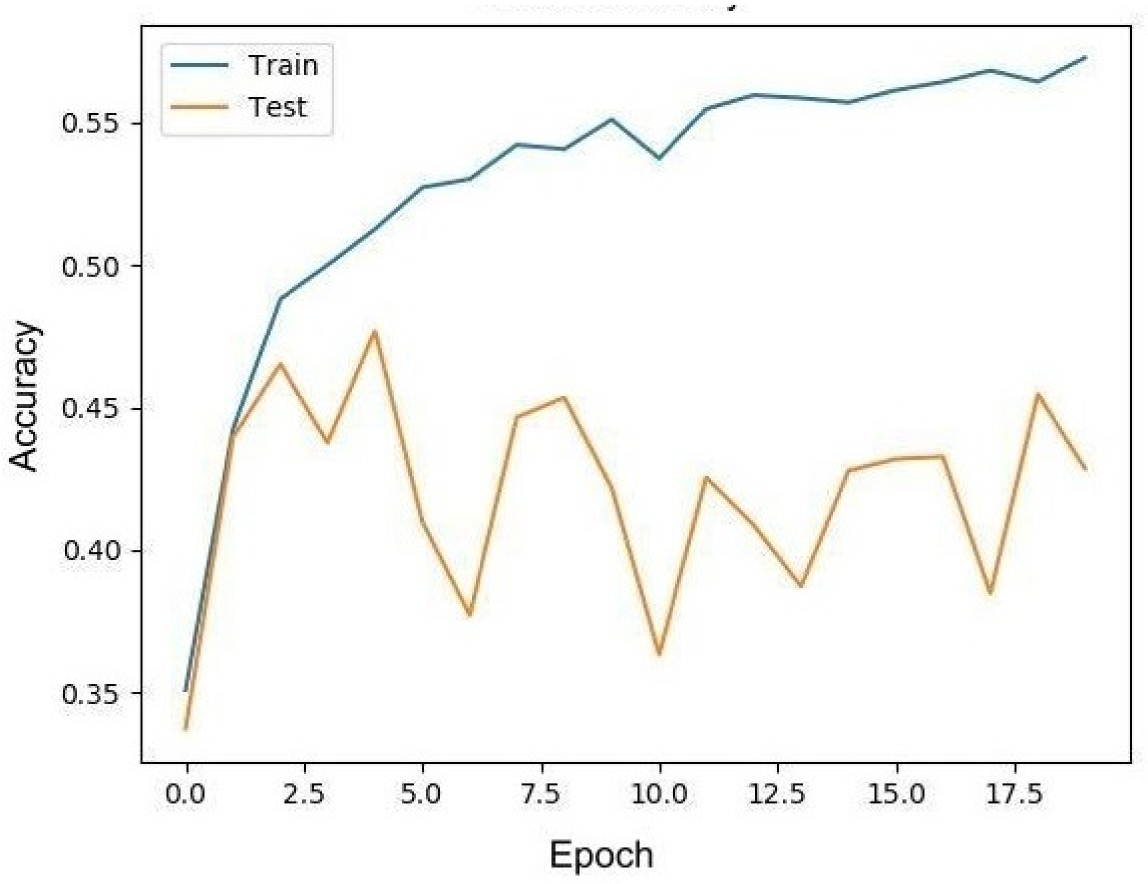

Figure 24. The training and testing history over the 20 epochs when using Inception model with the whole images.

cross validation, the hyperparameters were selected to have 50 epochs, using Adam optimizer, and using learning rate = 0.001.

The confusion matrix obtained by using this model is shown in Table 13. Also, The training and testing accuracy over the 50 epochs is shown in Figure 25. The training accuracy remained almost constant for the first 15 epochs then started to increased after that. The obtained testing accuracy is 80%. This accuracy outperforms the other pretrained architectures and gives more evidence that the WBCs images are not sharing so many features with ImageNet dataset.

Table 13. The confusion matrix obtained when using first customized model.

		True Label			
		EOSINOPHIL	LYMPHOCYTE	MONOCYTE	NEUTROPHIL
Predicted Label	EOSINOPHIL	448	30	126	116
	LYMPHOCYTE	32	530	8	11
	MONOCYTE	12	29	390	4
	NEUTROPHIL	131	31	96	493

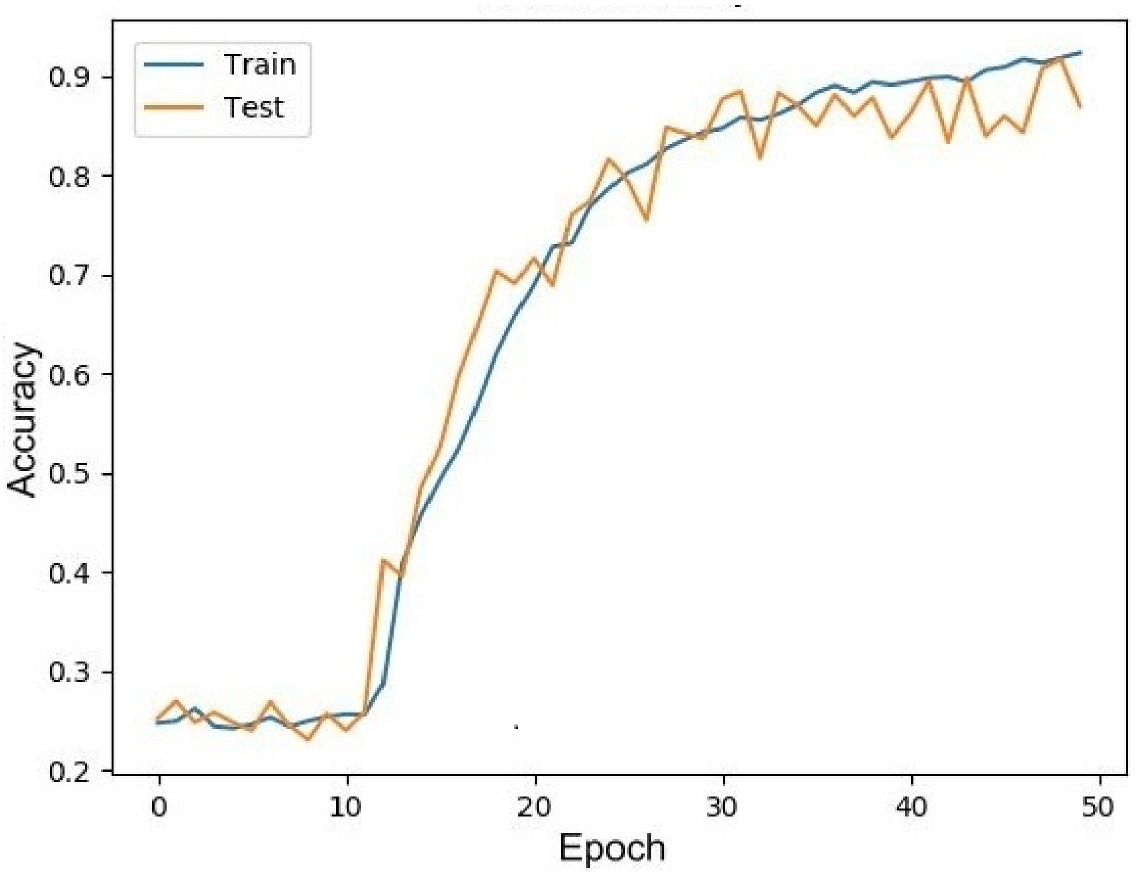

Figure 25. The training and testing history over the 50 epochs when using the first customized model with the whole images.

3.3.6 Classification using the second customized model with the whole images

In this experiment, the first customized model discussed in 2.2.2 is used. This model is also trained from scratch and it is more complicated than the first customized model. As a preprocessing steps, Images were resized for size of 60×60 and normalized to be between 0 and 1 by dividing all pixel values by 255. Using cross validation, the hyperparameters were selected to have 50 epochs, using 30 epochs, RMSprop optimizer and using learning rate $= 2 \times 10^{-5}$.

Using this model, the results outperformed all other models. The obtained accuracy is 89.5%. The confusion matrix obtained by using this model is shown in Table 14. Also, The training and testing accuracy over the 50 epochs is shown in Figure 26. This was the best result obtained when using the whole images for training and testing. In the next experiments, the effect of ROI extraction will be discussed.

3.4 The effect of ROI extraction on the classification results

In order to learn more important features and avoid having confusing features from the background that may affect the WBCs classification, another step is added to the pipeline, where the ROI containing the WBCs are extracted first prior to the classification stage.

From the two experiments discussed in 3.2.1 and 3.2.2, it was shown that the Mask RCNN algorithm using VGG pretrained network led to the best performance in the ROI extraction, accordingly, it was selected for ROI extraction prior to the classification. In the next subsection, the improvement due to the ROI extraction in both the pretrained and the customized architectures will be discussed.

3.4.1 The effect of ROI extraction on VGG architecture results

The VGG16 architecture achieved noticeable improvement in classification accuracy due to using the extracted ROI in the classification. When using the whole images, the achieved accuracy was 48.8%, while using the extracted ROI, the achieved accuracy is now 64%. The confusion matrix obained when using VGG16 with ROI extraction is shown in Table 15. It is obvoius that that the best results were obtained in LYMPHOCYTE class and EOSINOPHIL class respectively, while the MONOCYTE class showed less accuracy and the NEUTROPHIL class images were all classified in-

Table 14. The confusion matrix obtained when using second customized model.

		True Label			
		EOSINOPHIL	LYMPHOCYTE	MONOCYTE	NEUTROPHIL
Predicted Label	EOSINOPHIL	548	36	50	31
	LYMPHOCYTE	30	572	29	33
	MONOCYTE	32	4	488	18
	NEUTROPHIL	10	11	53	542

correctly. Most of the NEUTROPHIL class images were classified as EOSINOPHIL. The training and testing accuracy over the epochs are shown in Figure 27.

Table 15. The confusion matrix obtained when using VGG16 with the extracted ROI .

	True Label				
		EOSINOPHIL	LYMPHOCYTE	MONOCYTE	NEUTROPHIL
Predicted label	EOSINOPHIL	516	1	280	574
	LYMPHOCYTE	18	602	0	3
	MONOCYTE	24	3	252	9
	NEUTROPHIL	0	0	0	0

3.4.2 The effect of ROI extraction on ResNet architecture results

The ResNet architecture performance was improved significantly when using the extracted ROI. The accuracy increased from 74.6% to 84% due to the ROI extraction. The best accuracy achieved was in LYMPHOCYTE class, EOSINOPHIL class,

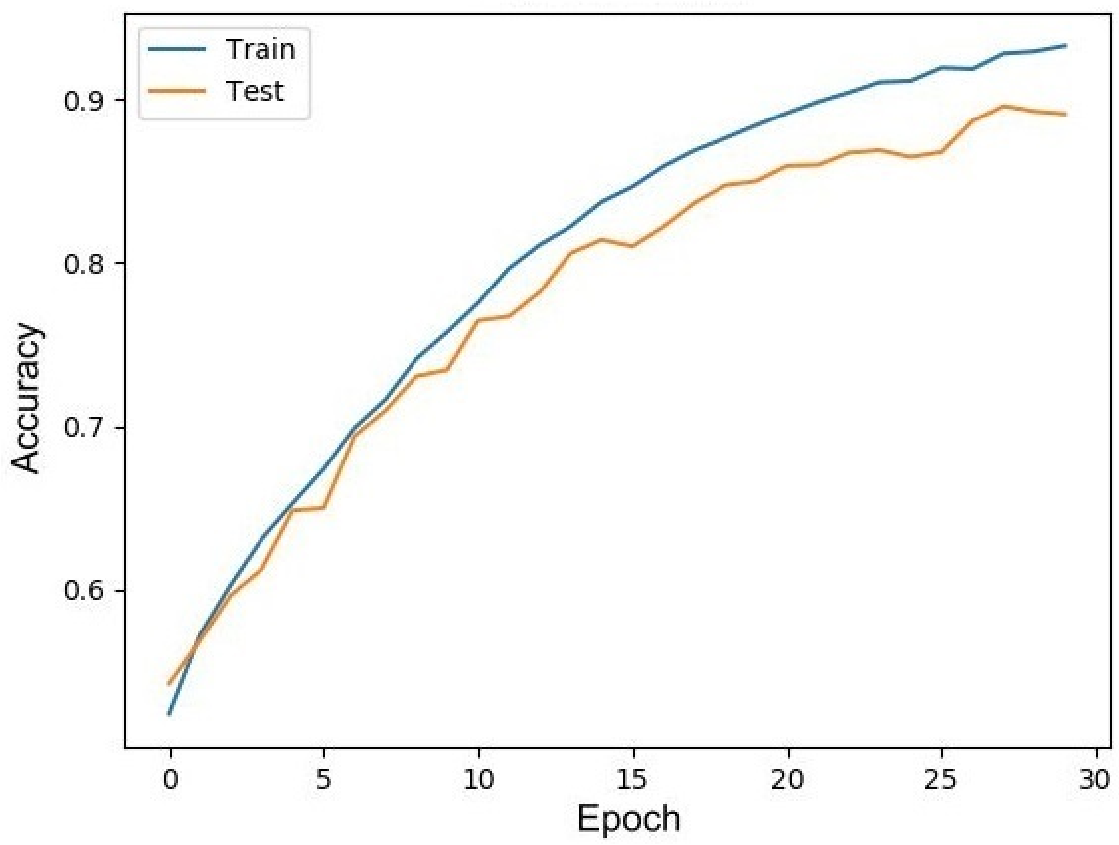

Figure 26. The training and testing history over the 30 epochs when using the second customized model with the whole images.

MONOCYTE class, then NEUTROPHIL class. The main source of error was in classifying NEUTROPHIL images as MONOCYTE class. The obtained confusion matrix is diplayed in Table 16. Also, the training and testing over the 20 epochs is diplayed in Figure 28

3.4.3 The effect of ROI extraction on Inception architecture results

Like the VGG and the ResNet, the inception architecture also achieved better results due to the ROI extraction. The classification accuracy was 43% when using the

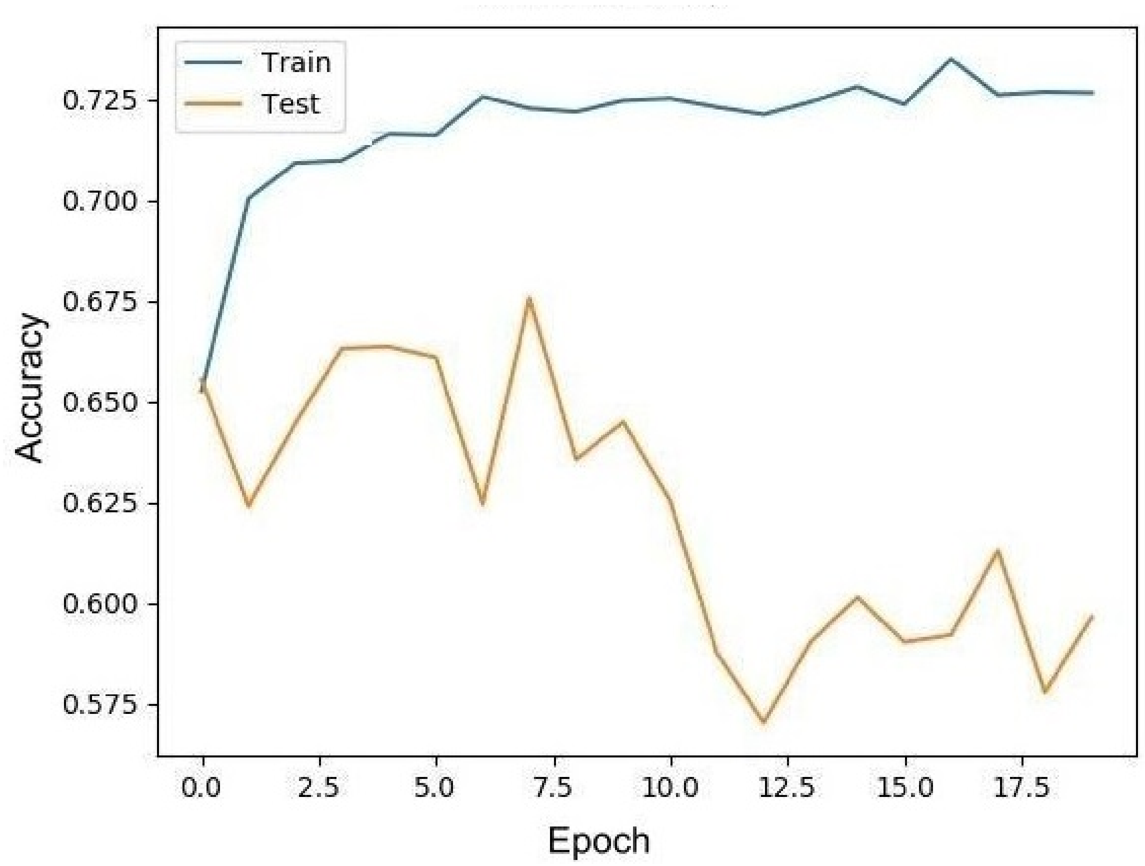

Figure 27. The training and testing history over the 20 epochs when using the VGG16 with the extracted ROI.

Table 16. The confusion matrix obtained when using ResNEt50 with the extracted ROI .

		True Label			
		EOSINOPHIL	LYMPHOCYTE	MONOCYTE	NEUTROPHIL
Predicted label	EOSINOPHIL	508	0	5	57
	LYMPHOCYTE	1	603	3	16
	MONOCYTE	0	3	438	90
	NEUTROPHIL	49	0	86	423

whole images, and it increased 55% due to the ROI extraction. However the ROI extraction was not able to resolve the main source of error that was generated due to the confusion between NEUTROPHIL and EOSINOPHIL classes. The confusion matrix displayed in Table 17shows that all most of the EOSINOPHIL was classified as NEUTROPHIL. Figure 29 shows the training and testing accuracy over the 20 epochs.

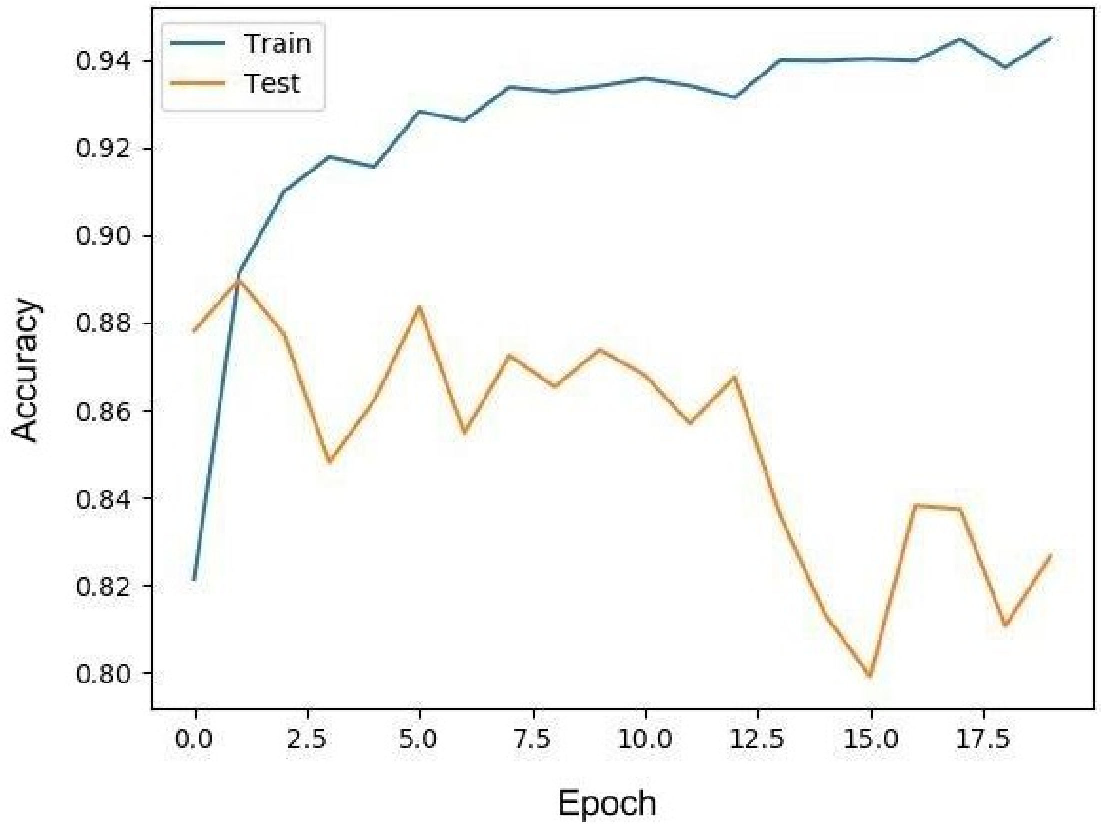

Figure 28. The training and testing history over the 20 epochs when using the ResNet50 with the extracted ROI.

Table 17. The confusion matrix obtained when using inception with the extracted ROI .

		True Label			
		EOSINOPHIL	LYMPHOCYTE	MONOCYTE	NEUTROPHIL
Predicted label	EOSINOPHIL	341	137	48	32
	LYMPHOCYTE	89	317	46	34
	MONOCYTE	41	84	322	186
	NEUTROPHIL	87	68	116	334

3.4.4 The effect of ROI extraction on the first customized architecture results

The first customized architecture achieved the largest improvement among all other architectures when using the extracted ROIs. The accuracy increased from 81% to 88%. The main source of error is having images from MONOCYTE class classified as NEUTROPHIL. The confusion matrix obtained from the first customized architecture when using the extracted ROIs is shown in Table 18. Also the training and testing accuracy over the 50 epochs are shown in figure 30.

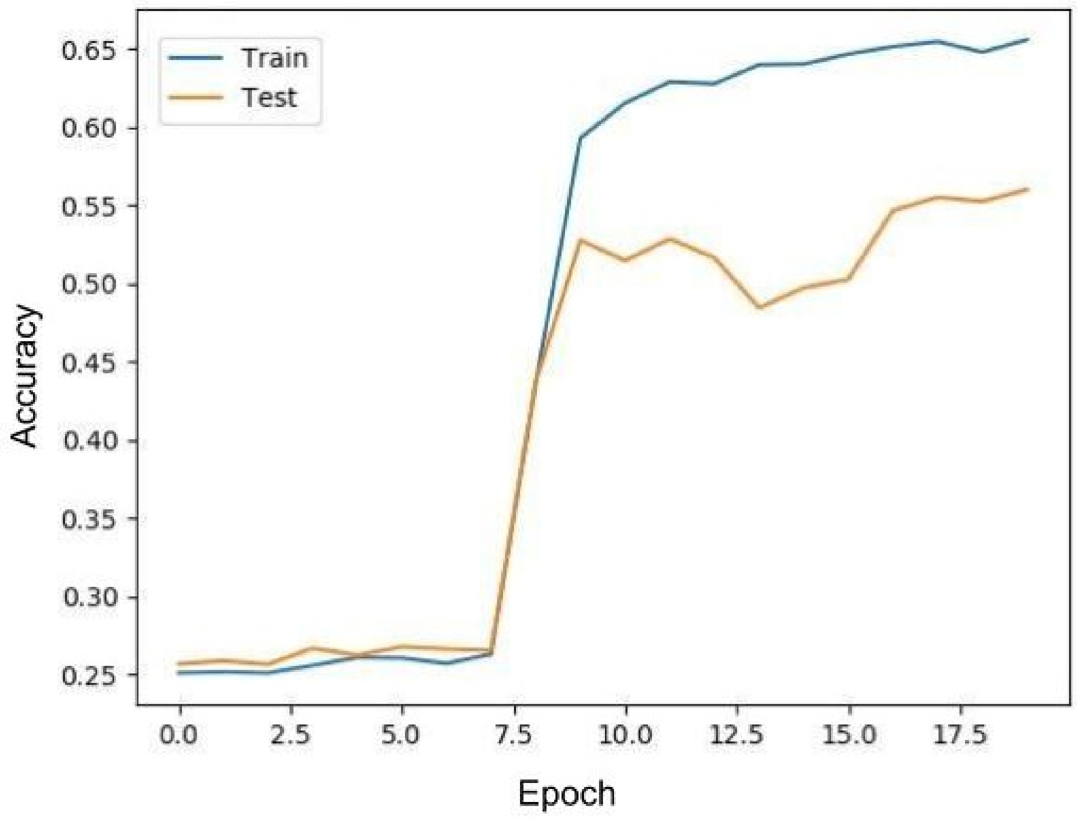

Figure 29. The training and testing history over the 20 epochs when using the Inception with the extracted ROI.

Table 18. The confusion matrix obtained when using the first customized CNN architecture with the extracted ROI .

		True Label			
		EOSINOPHIL	LYMPHOCYTE	MONOCYTE	NEUTROPHIL
Predicted label	EOSINOPHIL	509	0	0	49
	LYMPHOCYTE	0	603	2	0
	MONOCYTE	0	3	384	11
	NEUTROPHIL	49	0	146	526

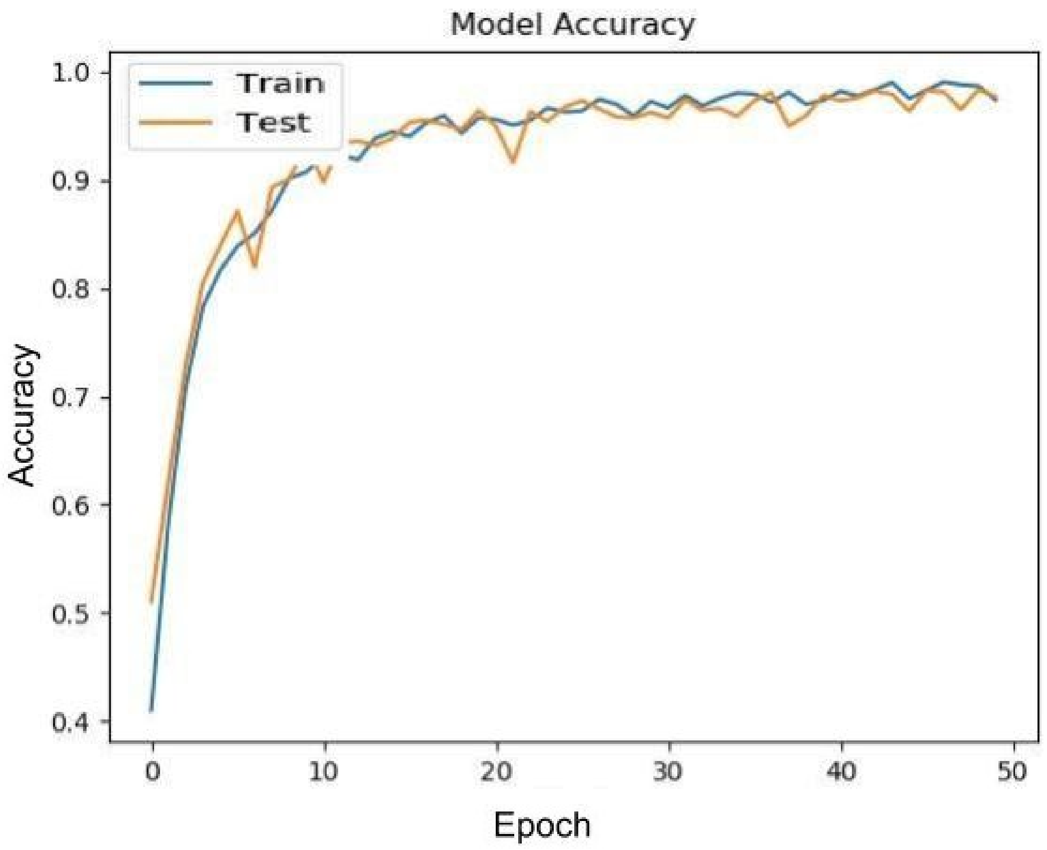

Figure 30. The training and testing history over the 50 epochs when using the first customized CNN architecture with the extracted ROI.

3.4.5 The effect of ROI extraction on the second customized architecture results

This architecture resulted in the best accuracy when using the whole images and it also resulted in the highest accuracy when using the extracted ROIs only. The accuracy increased from 89% to 92%. The main source of error is from classifying the LYMPHOCYTE as MONOCYTE images. The confusion matrix obtained from this experiment is displayed in Table 19. The training and testing accuracy over the 50 epochs are shown in Figure 26.

Since the customized network architecture with the extracted ROIs generated the best results, it will be the selected model in the next experiments. In the next section, the robustness of the model will be tested by adding noise to the testing images. The results will be reported by different types of noise at different signal to noise ratio (SNR).

Table 19. The confusion matrix obtained when using the second customized CNN architecture with the extracted ROI .

		True Label			
Predicted label		EOSINOPHIL	LYMPHOCYTE	MONOCYTE	NEUTROPHIL
	EOSINOPHIL	512	32	17	12
	LYMPHOCYTE	38	533	23	3
	MONOCYTE	5	26	483	45
	NEUTROPHIL	3	15	9	526

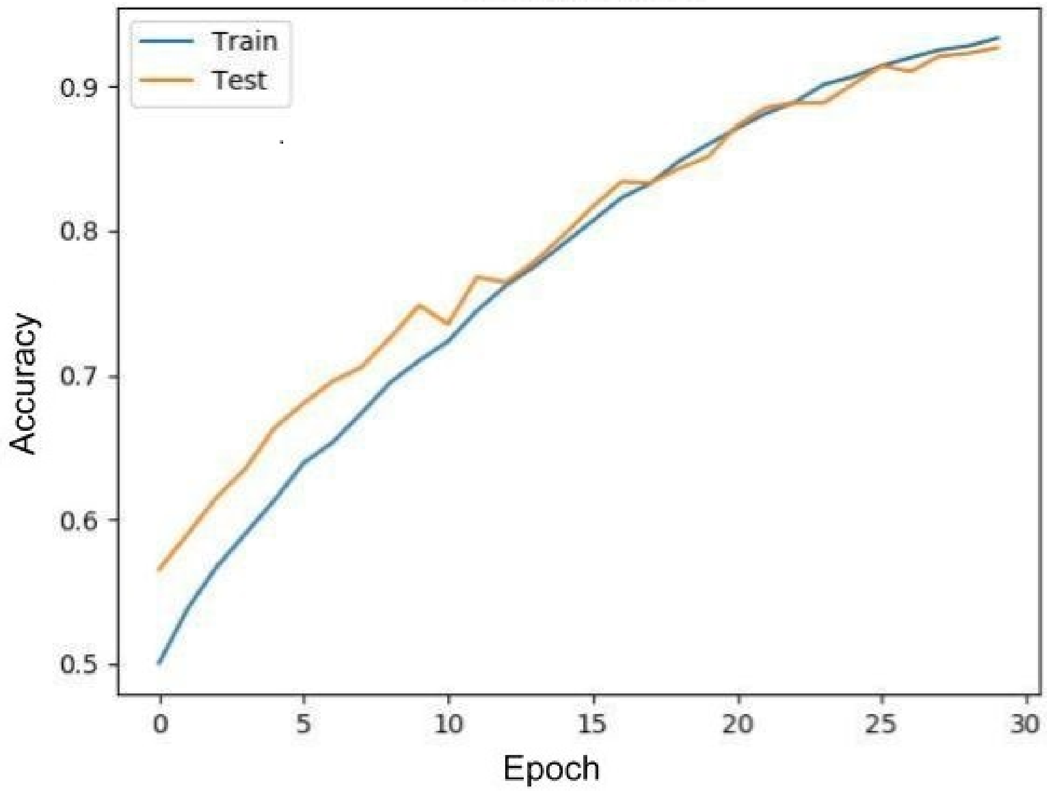

Figure 31. The training and testing history over the 30 epochs when using the second customized CNN architecture with the extracted ROI.

3.5 The effect of adding noise on the testing data

In this section, the effect of adding 3 types of noise will be discussed, (i) Gaussian noise, (ii) salt and pepper noise, and (iii) speckle noise. For each type of noise multiple noise levels will be used and the accuracy 1 will be reported at each level. The model to be used in these experiments is the second customized model as it was the model that yielded the highest accuracy on the testing data without adding noise.

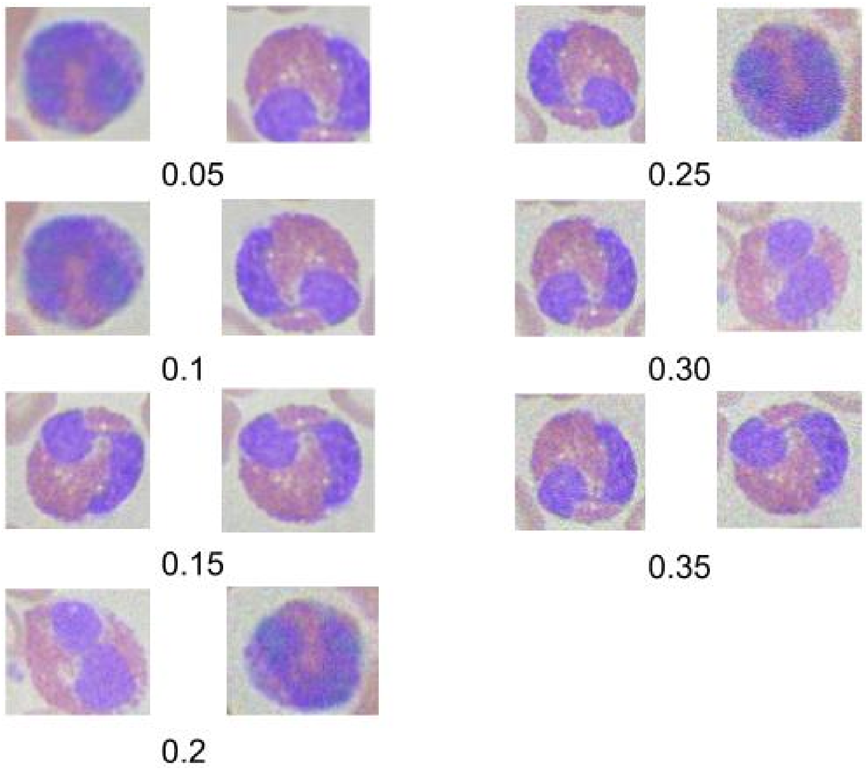

Figure 32. Sample of testing images with different levels of Gaussian noise added

3.5.1 The effect of adding Gaussian noise

To generate Gaussian noise, a random Gaussian noise with zero mean and variance = 0.01 with the same size as the original image. The noise image N is generated using the following equation:

$$N = I + G * \alpha \tag{1}$$

Where I is the original image, G is the Gaussian noise, and α is the noise factor that controls the SNR. The used α values are [0.05,0.1,0.15,0.2,0.25,0.3,0.35]. Figure 32 shows samples of the images at different α values. When using the images at different noise levels, the obtained accuracy started to drop as the noise level increases. At $\alpha = 0.05$ the obtained accuracy was 81% while at $\alpha = 0.35$ the obtained accuracy was 51%. Figure 33 shows the accuracy at different α levels.

3.5.2 The effect of adding salt and pepper noise

Salt and pepper noise is a kind of noise where an amount (α) of pixels in the image have high values (near 255), while another amount of the pixels (α) have low values (near 0). The noise is distributed uniformly random over the image space. [65]. In this experiment different values of α are used. The used values are: [0.004, 0.006, 0.008, 0.01, 0.012, 0.014, 0.016]. Figure 34 shows samples of images with salt and pepper noise at different α levels, and Figure 35 shows the model performance at

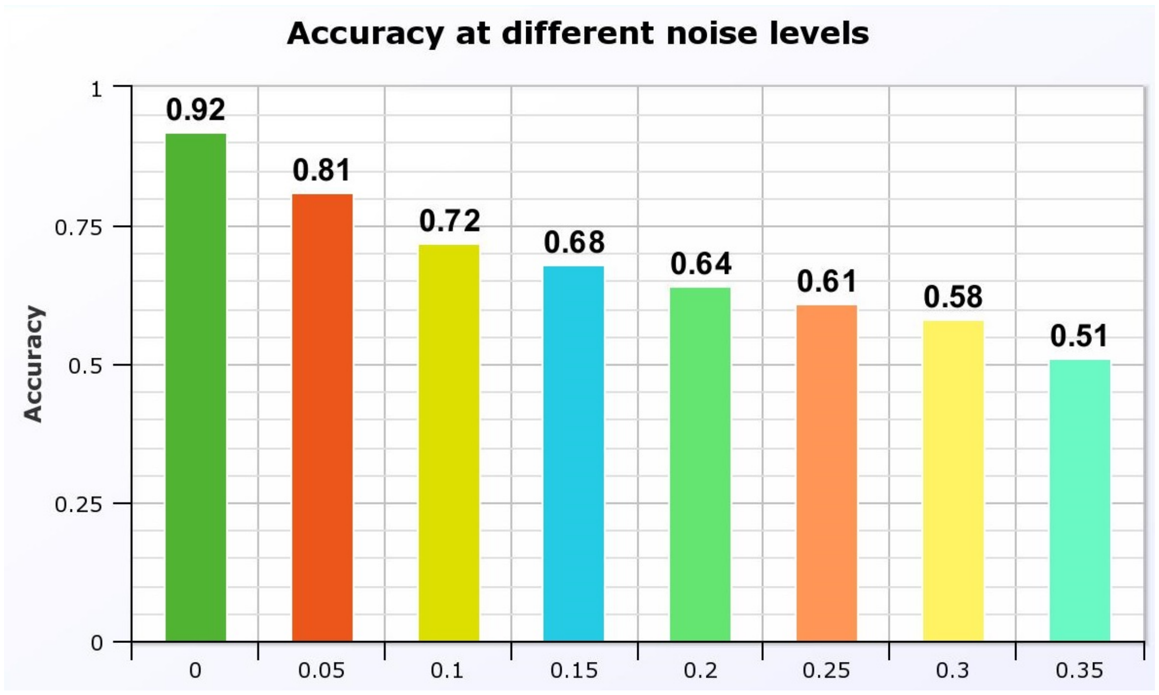

Figure 33. The obtained accuracy at different α levels for Gaussian noise.

each noise level. The model performance dropped to 69.5% at $\alpha = 0.004$ and to 59% at $\alpha = 0.016$.

3.5.3 The effect of adding speckle noise

Speckle is a granular interference that inherently exists in and degrades the quality of the images [66]. To generate the speckle noise, a random Gaussian noise G with the same size as the image is generated then the noisy image is given by:

$$N = I + I * G * \alpha \tag{2}$$

Where α controls the noise level and I is the original image. The used values for α were: [0.02, 0.04, 0.06, 0.08, 0.1, 0.12, 0.14]. Figure 36 shows samples of images at different noise levels and Figure 37 shows the model performance at each noise level. The accuracy at $\alpha = 0.02$ was 86% and it dropped 53% at $\alpha = 0.14$

3.6 Results summary discussion

In this chapter, four types of experiments were performed: (i) ROI extraction, (ii) classification of the white blood cells using the whole images, (iii) classification of the white blood cell using the extracted ROI, and (iv) the effect of the adding noise to the testing dataset.

For the ROI extraction, two approaches were tested, the color based approach and the Msk R-CNN. The evaluation metric used is the IoU. The best performance

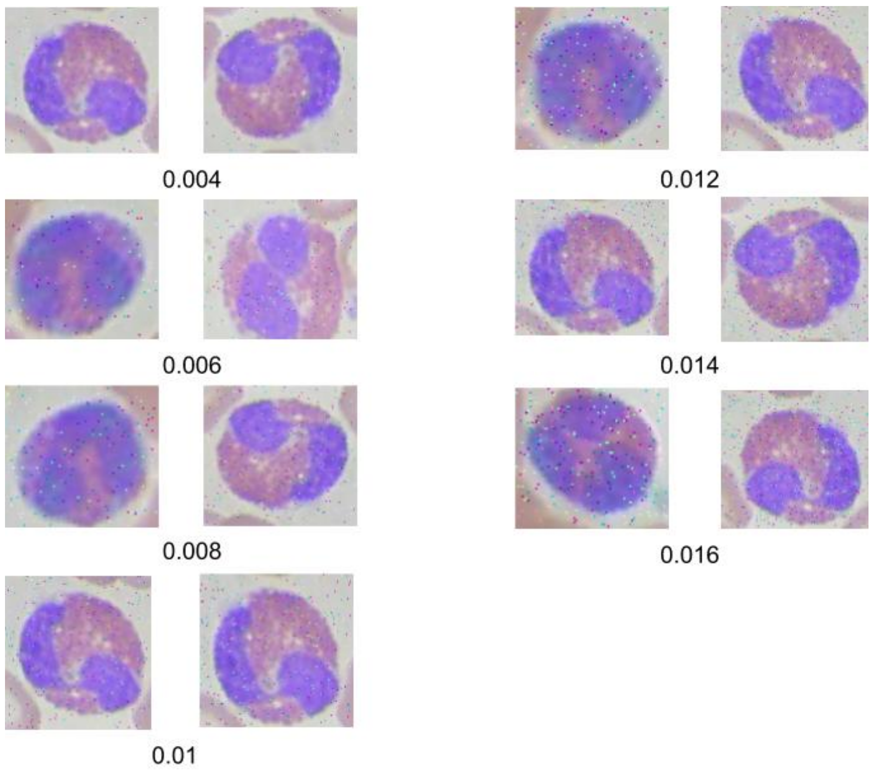

Figure 34. Sample of testing images with different levels of salt and pepper noise added

obtained is using the Mask R-CNN with the VGG pretrained architecture. Table 20 shows the summary statistics of the IoU when using both approaches. The mean IoU when using the Mask R-CNN is 0.71 with standard deviation 0.11, while the mean IoU when using the color based approach is 0.59 with standard deviation 0.12.

For the classification experiments, both pretrained and customized models were tested using the whole images and the extracted ROI images. The results indicated that the customized models outperformed the pretrained models with the whole images and with the ROIs. The results also showed significant improvement in the classification accuracy for all model when using only the extracted ROI. The results obtained from the different architectures for the whole images and for the extracted ROIs is shown in Table 21.

One reason for having the customized models out performing the pretrained models could be the big difference between the white blood cells images and the ImageNet dataset, this may reduce the number of useful features extracted through the pretrained networks. The performance improvements when using the extracted ROIs only could be justified by increasing the number of useful features and eliminating any shape features from the background that could make it more difficult for the models to learn.

In order the model robustness, the effect of adding noise to the testing images was also tested. The noise types used are Gaussian noise, salt and pepper noise,

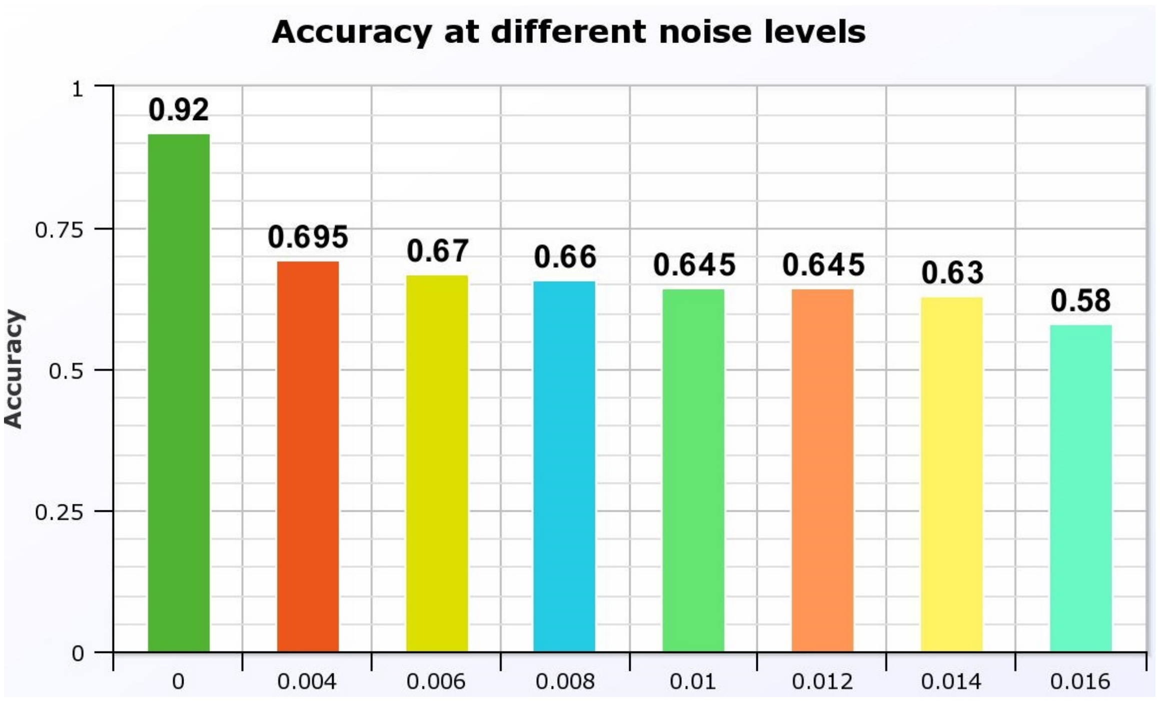

Figure 35. The obtained accuracy at different α levels using for salt and pepper noise.

and speckle noise. Each noise type was added with different amount to the testing dataset and the model performance was recorded. The accuracy drop in the model performance was proportional to the noise levels without any abrupt changes. This experiment showed that with reasonable noise amount, the model can still provide high classification accuracy.

The model performance was compared to other studies in the literature addressing the same problem. The comparison considered both the classification accuracy and the size of the dataset used. The suggested model is outperforming the other studies when taking into consideration the relatively large dataset used. The size of the dataset and the noise tolerance suggest that the proposed model will have good generalization ability. Table 22 shows a comparison between the proposed approach and some different approaches in the literature. The only approach that used larger dataset (13,000 sample) achieved accuracy 88% while the suggested approach achieves 92.5%, While the [33] achieved higher accuracy (94%), it was using very limited number of samples (84 samples).

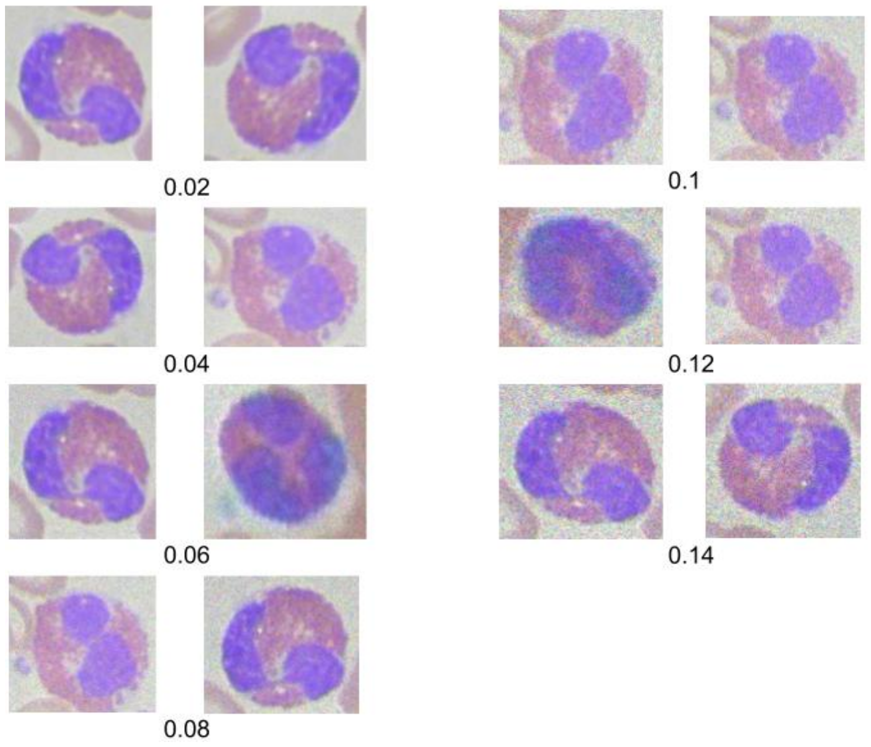

Figure 36. Sample of testing images with different levels of speckle noise added

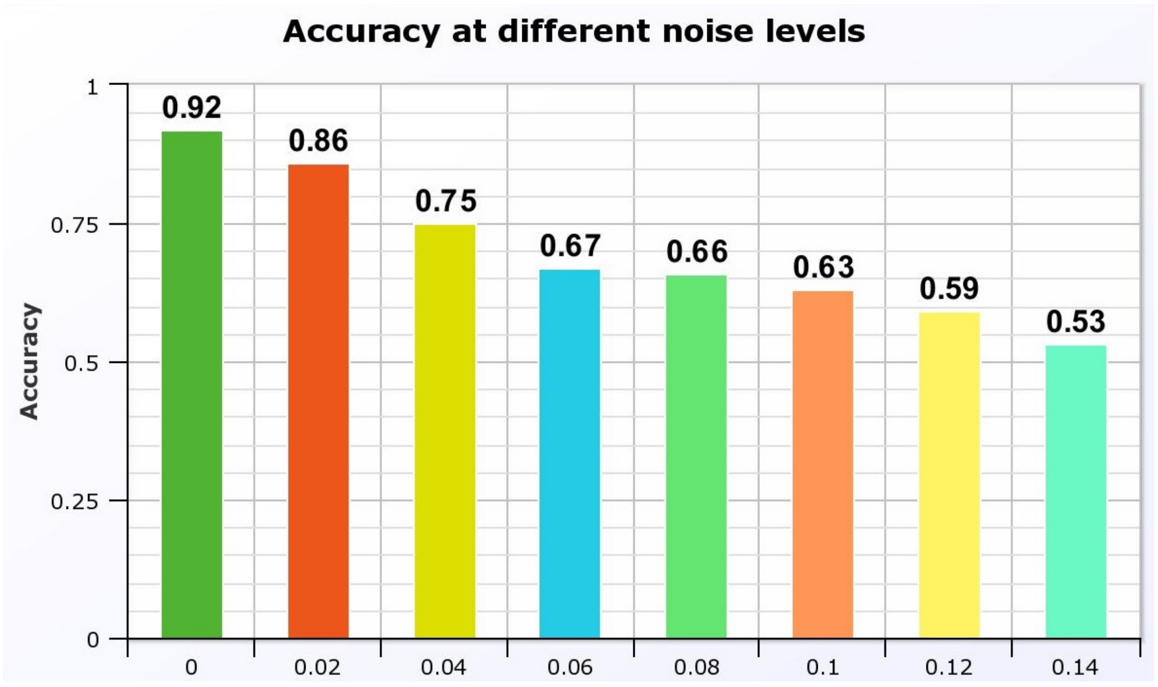

Figure 37. The obtained accuracy at different α levels using for speckle noise.

Table 20. Comparison of performance of both the Mask R-CNN and the color based approach for ROI extraction

	Mask R-CNN using VGG16	Color based Approach
Mean Iou	0.71	0.59
Minimum IoU	0.06	0.25
Maximum IoU	0.92	0.91
Standard deviation of IoU	0.11	0.12
Median IoU	0.73	0.6

Table 21. Comparison between different models performance with the whole images and with the extracted ROI. The best accuracy obtained was using the second customized model with the ROI images.

	Accuracy	
	Whole Images	Extracted ROI
VGG16	48.4%	64%
ResNet50	74.6%	84%
Inception	43%	55%
First customized model	80%	88%
Second customized model	89.5%	**92%**

Table 22. Comparison of some of the performing papers in the literature and the proposed approach.

Study	Number of samples	Accuracy
[32]	108	91%
[33]	84	94%
[45]	13,000	88%
[46]	263	92%
The proposed approach	**12,444**	**92.5%**

CHAPTER 4
CONCLUSIONS AND FUTURE WORK

In this work, a pipeline for white blood cells localization and classification is introduced. The white blood cells are divided into four main classes, (i) Neutrophils, (ii) Lymphocytes, (iii) Monocytes, and (iv) Eosinophils. The white blood cells plays important role in the human body, and the increase or decrease of the normal count of any type of the four types may lead to diseases like Leukocytosis, Autoimmune neutropenia, Severe congenital neutropenia, Cyclic neutropenia, Chronic granulomatous, and Leukocyte adhesion deficiencies (LAD syndrome).

The proposed pipeline consists of two main steps: (i) region of interest extraction, and (ii) classification of white blood cells. For each of the steps multiple experiments were conducted to find the optimal model performance. After finding the optimal model, the robustness of the pipeline was tested by adding three different types of noise: (i) Gaussian noise, (ii) salt and pepper noise, and (iii) speckle noise.

For the region of interest extraction, two approaches were tested, a classical approach using the color features of white blood cells and a deep learning based approach using Mask R-CNN. The data used for training the ROI component consists of 364 images with bounding box annotation coordinates given by experts for red and white blood cells which is used as ground truth for evaluation. The used metric for ROI extraction evaluation is the Intersection over Union (IoU). The average IoU obtained when using the classical approach is 60%, while the average IoU when using Mask R-CNN whit ResNet50 pretrained architecture is 69%, and with using VGG16 architecture is 71%. Based on these results, the selected ROI extraction algorithm is the Mask R-CNN with VGG16 pretrained network.

The second stage in the proposed pipeline is the classification step. In this step, both pretrained networks and customized models were tested. In addition, these architectures were tested with the whole images and with the extracted ROI only. The dataset used in the classification step consists of 12,444 images with 75% of the data for training and 25% for testing. The data is balanced in terms of the 4 classes. Each class represents approximately 25% of the training and testing datasets. The highest achieved accuracy is 92% and it was obtained using the second customized model with extracted RoI images.

In order to check the model robustness, another set of experiments were performed to study the effect of noise on the model accuracy. The 3 types of noise tested were

Gaussian noise, salt and pepper noise and speckle noise. For each of these noise types, different noise levels were added, and the accuracy was reported at each noise level. As the noise level increases, the accuracy decreases, but for all noise types and levels, there was no abrupt changes in the accuracy. With low noise levels, the accuracy drop was about 10% , while with very high noise levels, the lowest obtained accuracy was 59%.

The main contribution in this study is creating and validating a robust algorithm and testing it on relatively large dataset and with relatively high accuracy. Table 22 shows samples of the previous studies and their achieved results.

Although, the proposed model showed very good results, there are some limitation to be considered in the future work. One of these limitations is the need to have manually annotated images for initial training of the ROI extraction phase. This is only needed one time at the training phase, but is still could be time consuming task to do. To overcome this limitation, one possible approach is to develop an unsupervised learning technique that does not need ground truth for localization and use it for ROI extraction. Another limitation in this system appears when there are images with multiple white blood cells types in the same image. However should be already solved as the Mask R-CNN could extract multiple ROIs. We were not able to test this case in this dataset as the training data comes with one blood cell type and one label per image. We need another dataset with multiple classes per image to test the performance in this case. The third limitation comes from using deep learning models for classification. This makes the interpretation of the results much harder. Although we can visualize the extracted feature maps, it is still hard to trace the classification error and trace it. This is one of the most common drawback of using deep learning techniques.

In the future work, this system could be a part of a mobile application service that could be used for white blood cells localization and classification using cell phone camera. There has been already proposed systems for red blood cells counting using cell phone camera [67]. This could be a very good step to adapt such system to be also working on the white blood cells classification.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in "Proceedings of the IEEE conference on computer vision and pattern recognition," (2016), pp. 770–778.

[2] B. Blaus, "Medical gallery of blausen medical 2014," Wiki J Med **1**, 10 (2014).

[3] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," International journal of computer vision **115**, 211–252 (2015).

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

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556 (2014).

[6] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in "Proceedings of the IEEE conference on computer vision and pattern recognition," (2015), pp. 1–9.

[7] S. Siuly and Y. Zhang, "Medical big data: neurological diseases diagnosis through medical data analysis," Data Science and Engineering **1**, 54–64 (2016).

[8] G. Sinha, "Cad based medical image processing: Emphasis to breast cancer detection," i-Manager's Journal on Software Engineering **12**, 15 (2017).

[9] K. Abe, H. Takeo, Y. Nagai, Y. Kuroki, and S. Nawano, "Creation of new artificial calcification shadows for breast cancer and verification of effectiveness of cad development technique that uses no actual cases," in "14th International Workshop on Breast Imaging (IWBI 2018)," , vol. 10718 (International Society for Optics and Photonics, 2018), vol. 10718, p. 1071817.

[10] D. Sathish, S. Kamath, K. Rajagopal, and K. Prasad, "Medical imaging techniques and computer aided diagnostic approaches for the detection of breast cancer with an emphasis on thermography-a review," International Journal of Medical Engineering and Informatics **8**, 275–299 (2016).

[11] W. K. Moon, I.-L. Chen, J. M. Chang, S. U. Shin, C.-M. Lo, and R.-F. Chang, "The adaptive computer-aided diagnosis system based on tumor sizes for the classification of breast tumors detected at screening ultrasound," Ultrasonics **76**, 70–77 (2017).

[12] M. Liang, W. Tang, D. M. Xu, A. C. Jirapatnakul, A. P. Reeves, C. I. Henschke, and D. Yankelevitz, "Low-dose ct screening for lung cancer: computer-aided detection of missed lung cancers," Radiology **281**, 279–288 (2016).

[13] A. Chon, N. Balachandar, and P. Lu, "Deep convolutional neural networks for lung cancer detection," Standford University (2017).

[14] R. Campa, M. Del Monte, G. Barchetti, M. Pecoraro, V. Salvo, I. Ceravolo, E. L. Indino, A. Ciardi, C. Catalano, and V. Panebianco, "Improvement of prostate cancer detection combining a computer-aided diagnostic system with trus-mri targeted biopsy," Abdominal Radiology **44**, 264–271 (2019).

[15] G. Lemaitre, R. Martí, M. Rastgoo, and F. Mériaudeau, "Computer-aided detection for prostate cancer detection based on multi-parametric magnetic resonance imaging," in "2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)," (IEEE, 2017), pp. 3138–3141.

[16] V. Giannini, S. Rosati, D. Regge, and G. Balestra, "Specificity improvement of a cad system for multiparametric mr prostate cancer using texture features and artificial neural networks," Health and Technology **7**, 71–80 (2017).

[17] T. J. Brinton, Z. Ali, C. Di Mario, J. Hill, R. Whitbourn, M. Gotberg, U. Illindala, A. Maehara, N. Van Mieghem, I. Meredith *et al.*, "Performance of the lithoplasty system in treating calcified coronary lesions prior to stenting: results from the disrupt cad oct sub-study," Journal of the American College of Cardiology **69**, 1121 (2017).

[18] M. Usman, M. M. Fraz, and S. A. Barman, "Computer vision techniques applied for diagnostic analysis of retinal oct images: a review," Archives of Computational Methods in Engineering **24**, 449–465 (2017).

[19] R. Rasti, H. Rabbani, A. Mehridehnavi, and F. Hajizadeh, "Macular oct classification using a multi-scale convolutional neural network ensemble," IEEE transactions on medical imaging **37**, 1024–1034 (2017).

[20] S. Charfi, M. El Ansari, and I. Balasingham, "Computer-aided diagnosis system for ulcer detection in wireless capsule endoscopy images," IET Image Processing **13**, 1023–1030 (2019).

[21] K. K. Jani, S. Srivastava, and R. Srivastava, "Computer aided diagnosis system for ulcer detection in capsule endoscopy using optimized feature set," Journal of Intelligent & Fuzzy Systems pp. 1–8 (2019).

[22] B. García-Zapirain, M. Elmogy, A. El-Baz, and A. S. Elmaghraby, "Classification of pressure ulcer tissues with 3d convolutional neural network," Medical & biological engineering & computing **56**, 2245–2258 (2018).

[23] M.-C. Su, C.-Y. Cheng, and P.-C. Wang, "A neural-network-based approach to white blood cell classification," The scientific world journal **2014** (2014).

[24] B. Shopsin, R. Friedmann, and S. Gershon, "Lithium and leukocytosis," Clinical Pharmacology & Therapeutics **12**, 923–928 (1971).

[25] L. A. Boxer, M. S. Greenberg, G. J. Boxer, and T. P. Stossel, "Autoimmune neutropenia," New England Journal of Medicine **293**, 748–753 (1975).

[26] K. Devriendt, A. S. Kim, G. Mathijs, S. G. Frints, M. Schwartz, J. J. Van den Oord, G. E. Verhoef, M. A. Boogaerts, J.-P. Fryns, D. You *et al.*, "Constitutively activating mutation in wasp causes x-linked severe congenital neutropenia," Nature genetics **27**, 313–317 (2001).

[27] D. C. Dale, A. A. Bolyard, and A. Aprikyan, "Cyclic neutropenia," in "Seminars in hematology," , vol. 39 (Elsevier, 2002), vol. 39, pp. 89–94.

[28] R. L. Baehner and D. G. Nathan, "Quantitative nitroblue tetrazolium test in chronic granulomatous disease," New England Journal of Medicine **278**, 971–976 (1968).

[29] T. W. Kuijpers, R. Van Lier, D. Hamann, M. de Boer, L. Y. Thung, R. S. Weening, A. J. Verhoeven, and D. Roos, "Leukocyte adhesion deficiency type 1 (lad-1)/variant. a novel immunodeficiency syndrome characterized by dysfunctional beta2 integrins." The Journal of clinical investigation **100**, 1725–1733 (1997).

[30] A. Tefferi, C. A. Hanson, and D. J. Inwards, "How to interpret and pursue an abnormal complete blood cell count in adults," in "Mayo Clinic Proceedings," , vol. 80 (Elsevier, 2005), vol. 80, pp. 923–936.

[31] V. Marchesi and J. L. Gowans, "The migration of lymphocytes through the endothelium of venules in lymph nodes: an electron microscope study," Proceedings of the Royal Society of London. Series B. Biological Sciences **159**, 283–290 (1964).

[32] G. Ongun, U. Halici, K. Leblebicioglu, V. Atalay, M. Beksaç, and S. Beksaç, "Feature extraction and classification of blood cells for an automated differential blood count system," in "IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222)," , vol. 4 (IEEE, 2001), vol. 4, pp. 2461–2466.

[33] N. Sinha and A. Ramakrishnan, "Automation of differential blood count," in "TENCON 2003. Conference on Convergent Technologies for Asia-Pacific Region," , vol. 2 (IEEE, 2003), vol. 2, pp. 547–551.

[34] C. Hu, L.-J. Jiang, and J. Bo, "Wavelet transform and morphology image segmentation algorism for blood cell," in "2009 4th IEEE Conference on Industrial Electronics and Applications," (IEEE, 2009), pp. 542–545.

[35] A. Shahin, Y. Guo, K. Amin, and A. A. Sharawi, “A novel white blood cells segmentation algorithm based on adaptive neutrosophic similarity score,” Health information science and systems **6**, 1 (2018).

[36] Y. Guo, A. Şengür, and J. Ye, “A novel image thresholding algorithm based on neutrosophic similarity score,” Measurement **58**, 175–186 (2014).

[37] F. Sadeghian, Z. Seman, A. R. Ramli, B. H. A. Kahar, and M.-I. Saripan, “A framework for white blood cell segmentation in microscopic blood images using digital image processing,” Biological procedures online **11**, 196 (2009).

[38] F. Liu, B. Zhao, P. K. Kijewski, L. Wang, and L. H. Schwartz, “Liver segmentation for ct images using gvf snake,” Medical physics **32**, 3699–3706 (2005).

[39] G. W. Zack, W. E. Rogers, and S. Latt, “Automatic measurement of sister chromatid exchange frequency.” Journal of Histochemistry & Cytochemistry **25**, 741–753 (1977).

[40] R. B. Hegde, K. Prasad, H. Hebbar, and B. M. K. Singh, “Development of a robust algorithm for detection of nuclei and classification of white blood cells in peripheral blood smear images,” Journal of medical systems **42**, 110 (2018).

[41] A. Shahin, Y. Guo, K. M. Amin, and A. A. Sharawi, “White blood cells identification system based on convolutional deep neural learning networks,” Computer methods and programs in biomedicine **168**, 69–80 (2019).

[42] H. Kutlu, E. Avci, and F. Özyurt, “White blood cells detection and classification based on regional convolutional neural networks,” Medical hypotheses **135**, 109472 (2020).

[43] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in “European conference on computer vision,” (Springer, 2016), pp. 21–37.

[44] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, “Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size,” arXiv preprint arXiv:1602.07360 (2016).

[45] P. Tiwari, J. Qian, Q. Li, B. Wang, D. Gupta, A. Khanna, J. J. Rodrigues, and V. H. C. de Albuquerque, “Detection of subtype blood cells using deep learning,” Cognitive Systems Research **52**, 1036–1044 (2018).

[46] Y. Y. Baydilli and Ü. Atila, “Classification of white blood cells using capsule networks,” Computerized Medical Imaging and Graphics p. 101699 (2020).

[47] S. Sabour, N. Frosst, and G. E. Hinton, “Dynamic routing between capsules,” in “Advances in neural information processing systems,” (2017), pp. 3856–3866.

[48] T. Tran, O.-H. Kwon, K.-R. Kwon, S.-H. Lee, and K.-W. Kang, "Blood cell images segmentation using deep learning semantic segmentation," in "2018 IEEE International Conference on Electronics and Communication Engineering (ICECE)," (IEEE, 2018), pp. 13–16.

[49] A. Tareef, Y. Song, D. Feng, M. Chen, and W. Cai, "Automated multi-stage segmentation of white blood cells via optimizing color processing," in "2017 IEEE 14th international symposium on Biomedical imaging (ISBI 2017)," (IEEE, 2017), pp. 565–568.

[50] S. J. Pan and Q. Yang, "A survey on transfer learning," IEEE Transactions on knowledge and data engineering **22**, 1345–1359 (2009).

[51] L. Shao, F. Zhu, and X. Li, "Transfer learning for visual categorization: A survey," IEEE transactions on neural networks and learning systems **26**, 1019–1034 (2014).

[52] K. Gopalakrishnan, S. K. Khaitan, A. Choudhary, and A. Agrawal, "Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection," Construction and Building Materials **157**, 322–330 (2017).

[53] H. Ide and T. Kurita, "Improvement of learning for cnn with relu activation by sparse regularization," in "2017 International Joint Conference on Neural Networks (IJCNN)," (IEEE, 2017), pp. 2684–2691.

[54] S. Mittal, "A survey of fpga-based accelerators for convolutional neural networks," Neural computing and applications pp. 1–31 (2018).

[55] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," arXiv preprint arXiv:1502.03167 (2015).

[56] F. Chen, N. Chen, H. Mao, and H. Hu, "Assessing four neural networks on handwritten digit recognition dataset (mnist)," arXiv preprint arXiv:1811.08278 (2018).

[57] K. Liu, M. Zhang, and Z. Pan, "Facial expression recognition with cnn ensemble," in "2016 international conference on cyberworlds (CW)," (IEEE, 2016), pp. 163–166.

[58] S. Suzuki *et al.*, "Topological structural analysis of digitized binary images by border following," Computer vision, graphics, and image processing **30**, 32–46 (1985).

[59] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in "Proceedings of the IEEE international conference on computer vision," (2017), pp. 2961–2969.

[60] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in "Proceedings of the IEEE conference on computer vision and pattern recognition," (2014), pp. 580–587.

[61] R. Faster, "Towards real-time object detection with region proposal networks shaoqing ren," Kaiming He, Ross Girshick, and Jian Sun (2015).

[62] R. Girshick, "Fast r-cnn," in "Proceedings of the IEEE international conference on computer vision," (2015), pp. 1440–1448.

[63] V. S. Mohan, R. Vinayakumar, V. Sowmya, and K. Soman, "Deep rectified system for high-speed tracking in images," Journal of Intelligent & Fuzzy Systems **36**, 1957–1965 (2019).

[64] Y. Wang, J. Liu, J. Mišić, V. B. Mišić, S. Lv, and X. Chang, "Assessing optimizer impact on dnn model sensitivity to adversarial examples," IEEE Access **7**, 152766–152776 (2019).

[65] R. J. Schalkoff, *Digital image processing and computer vision*, vol. 286 (Wiley New York, 1989).

[66] M. P. Wachowiak, A. S. Elmaghraby, R. Smolikova, and J. M. Zurada, "Classification and estimation of ultrasound speckle noise with neural networks," in "Proceedings IEEE International Symposium on Bio-Informatics and Biomedical Engineering," (IEEE, 2000), pp. 245–252.

[67] H. Zhu, I. Sencan, J. Wong, S. Dimitrov, D. Tseng, K. Nagashima, and A. Ozcan, "Cost-effective and rapid blood analysis on a cell-phone," Lab on a Chip **13**, 1282–1288 (2013).

CURRICULUM VITA

NAME: Omar Dekhil
Education:

- BSc., Electrical Engineering,
  Alexandria University,
  2006-2011
- MSc., Computer Informatics,
  Nile University,
  2013-2016
- PhD., Computer Science and Engineering
  University of Louisville
  2016-2020

AWARDS: Grosscurth Fellowship, university of Louisville

PROFESSIONAL SOCIETIES: IEEE, EMBS
PUBLICATIONS:

- Automatic localization of the left ventricle in cardiac MRI images using deep learning
- A novel CAD system for autism diagnosis using structural and functional MRI
- Using resting state functional MRI to build a personalized autism diagnosis system
- Identifying personalized autism related impairments using resting functional MRI and ADOS reports
- A Personalized Autism Diagnosis CAD System Using a Fusion of Structural MRI and Resting-State Functional MRI Data
- Towards Personalized Autism Diagnosis: Promising Results
- Autism Spectrum Disorder Diagnosis framework using Diffusion Tensor Imaging
- Deep Learning Based Method for Computer Aided Diagnosis of Diabetic Retinopathy

- A Machine Learning Approach for Grading Autism Severity Levels Using Task-based Functional MRI
- Functional Magnetic Resonance Imaging Based Framework for Autism Diagnosis
- A Novel Fully Automated CAD System for Left Ventricle Volume Estimation