Clinical Decision Support System with Adaptive Software Framework for Chronic Lymphocytic Leukaemia Cell Classification

by

Emad Amin Mohammed

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

CALGARY, ALBERTA

SEPTEMBER, 2013

© Emad Amin Mohammed 2013
Abstract
This thesis presents a new clinical decision support system (CDSS), which operates within an adaptive software framework and a tailored wrapper design pattern for chronic lymphocytic leukaemia (CLL) cell classification. The system goes through a sequence of steps while working with the lymphocyte images: it segments the lymphocyte with average segmentation accuracy of (97% ±0.5 for lymphocyte nucleus and 92.08% ±9.24 for lymphocyte cytoplasm); it extracts features; it selects from those features the relevant ones; and, it then classifies the selected features. The proposed system composite classifier model has a trust factor of 84.16%, accuracy of 87.0%, 84.95% true positive rate, and 10.96% false positive rate.

The framework along with the wrapper pattern became a generic interface for any new algorithm. The framework built on top of the data-centric architecture which provides a great flexibility to the system design. The wrapper verifies the new algorithm interface against built-in test procedures.
Preface

This preface lists the publications by the author of this thesis which includes the materials and the ideas presented in this thesis.


2. E. A. Mohammed, M. M. A. Mohamed, C. Naugler and B. H. Far, “Chronic lymphocytic leukemia cell segmentation using a machine learning algorithm,” Poster presentation, 27th World Congress of the World Association of Societies of Pathology and Laboratory Medicine (WASPaLM), Quebec City, PQ, 9 June 2013. Accepted and a poster is presented.


Acknowledgements

I would like to express my deep appreciation to my supervisor Dr. Behrouz Far and to my co-supervisor Dr. Christopher Naugler whose help are highly esteemed and this work could not have been done without their useful help, support, and encouragement. Dr. Far kept me aware of the fact that a Master of Science in Software Engineering is not only implementing algorithms, designing and coding software applications, but also a research process towards software patterns, frameworks and software architectures. Dr. Far kept telling me that “to make sure you are on the right track; take one step back and try one more time”. Particularly, I would like to express my gratitude to my co-supervisor Dr. Naugler who provided the data used to design and validate this research work, and spent a great deal of his time teaching me the nature of the problem, manually classifying the images and reviewing the results.

I would like to thank SmartLabLtd and MITACS accelerate program that provided me with financial support for the period of my internship, and I would like to thank Dr. Mostafa Mohammed for his technical advices and support.

I would like to show my sincere appreciation to Dr. Mohamed Z. Bendjaballah who discovered my potential, encouraged me to purse my graduate studies, and kept encouraging me till the moment. Dr. Bendjaballah kept telling me that “the pain I feel today will be the strength I will feel tomorrow”. Moreover I would like to be grateful to Dr. Mohammed I. Hashim for his endless help, and support. Dr. Hashim is always there for me, supporting, encouraging, and providing extraordinary help.

Finally, I thank the Department of Electrical and Computer Engineering at the University of Calgary for providing the administrative support.
Dedication

I dedicate my thesis work to many friends.

A special feeling of appreciation is dedicated to my mother, father, brother, sisters, and to my children.

To my wife

who has supported me all the way since the beginning of my studies,

and whose image needs no enhancements
Table of Contents

Abstract ........................................................................................................................................ ii
Preface ........................................................................................................................................ iii
Acknowledgements ................................................................................................................... iv
Dedication ................................................................................................................................. v
Table of Contents .................................................................................................................... vi
List of Tables ............................................................................................................................ x
List of Figures .......................................................................................................................... xiii
List of Symbols, Abbreviations, and Nomenclature .............................................................. xix

Chapter 1: Introduction ......................................................................................................... 1
  1.1 Motivations .................................................................................................................... 4
  1.2 Research Objectives ................................................................................................... 5
  1.3 Research novelty and significance ............................................................................ 7
  1.4 Research Contributions ............................................................................................ 9
  1.5 Research Methodology ............................................................................................. 10
  1.6 Structure of the Thesis ............................................................................................ 11

Chapter 2: Literature Review and Related Works ............................................................. 12
  2.1 Literature Review : CLL and CLL staging ............................................................... 12
  2.2 Literature Review : Clinical Decision Support System (CDSS) ............................ 15
  2.3 Literature Review : Microscopic Image Segmentation .......................................... 19
  2.4 Literature Review : Feature Extraction and Feature Selection ............................. 22
  2.5 Literature Review : Classification and Multiple Classifier System (MCS) .......... 26
      2.5.1 Artificial Neural Network (ANN) ...................................................................... 26
2.5.2 Support Vector Machine (SVM)........................................................................... 29
2.5.3 Decision tree ................................................................................................... 32
2.5.4 K-Nearest Neighborhood (K-NN) ............................................................... 33
2.5.5 Adaptive Boosting (AdaBoost) ................................................................. 33
2.5.6 Multiple Classifier System (MCS).............................................................. 35

2.6 Related work: Using MLA and MCS to classify the WBCs............................... 37

2.7 Literature Review: Software Architecture, Framework, and Design pattern... 39

2.8 Conclusion......................................................................................................... 45

Chapter 3: The Proposed System............................................................................ 48

3.1 Lymphocyte Images .......................................................................................... 50

3.2 Lymphocyte cell segmentation using a watershed algorithm and the Otsu’s threshold method ................................................................. 51

3.2.1 Nucleus Segmentation ................................................................................ 53
3.2.2 Cell Segmentation ...................................................................................... 54
3.2.3 Cytoplasm Segmentation ......................................................................... 55

3.3 Lymphocyte cell segmentation using MLA and K-means algorithm .......... 55

3.3.1 Nucleus Segmentation .............................................................................. 55
3.3.2 Cell Segmentation ..................................................................................... 58
3.3.3 Cytoplasm Segmentation ........................................................................ 59

3.4 Accuracy Measurement .................................................................................. 59

3.5 Feature extraction and feature selection algorithms ...................................... 60

3.6 Multiple Classifier System ............................................................................ 64

3.6.1 SVM classifier.......................................................................................... 66
5.5.1 Classifier models training performance .................................................... 119
5.5.2 The SVM classifier ................................................................................... 123
5.5.3 ANN classifier .......................................................................................... 126
5.5.4 KNN classifier .......................................................................................... 127
5.5.5 Decision Tree classifier ............................................................................. 128
5.5.6 AdaBoost classifier ................................................................................... 130
5.5.7 Fusion of the Classifier models results ..................................................... 131
5.5.8 Validation .................................................................................................. 137
5.6 System output ................................................................................................... 139
5.7 Conclusion........................................................................................................ 140

Chapter 6: Conclusion, Limitations, and Future Works ................................. 143
6.1 Conclusion ........................................................................................................ 143
6.2 Limitations ....................................................................................................... 147
6.3 The Future works – The Big Picture – ........................................................... 149

REFERENCES ............................................................................................................... 151
## List of Tables

Table 2.1 – Binet classification system for CLL ............................................................. 14
Table 2.2 – Rai and modified Rai classification staging system for CLL ....................... 14
Table 2.3 – Examples of CDSS interventions by target area of care.............................. 18
Table 2.4 – Measurable features of an object. .................................................................. 24
Table 2.5 – Common Software Architecture quality attributes and their definitions....... 42
Table 2.6 – The techniques used to develop the proposed system components with the reasons for using these techniques and the related thesis chapters.......................... 47
Table 3.1 – The images used to develop the proposed system ....................................... 51
Table 3.2 – Features and features interpretation used in this study............................... 62
Table 4.1 – Functional requirements of the proposed CDSS ......................................... 88
Table 4.2 – The proposed software framework component to achieve the hosting system requirements. ............................................................................................................. 89
Table 5.1 – Segmentation accuracy of the watershed algorithm using two different metrics: pixel counting and closed contour area overlapping (Cyto=Cytoplasm). ................................................................................................................................. 107
Table 5.2 – The Percentage segmentation accuracy measurement for 140 lymphocyte images segmented by the ANN algorithm which is evaluated by two different methods: Pixel counting and Area overlapping (Cyto=Cytoplasm). ....................... 109
Table 5.3 – The Percentage segmentation accuracy measurement for 140 lymphocyte images segmented by the SVM algorithm which is evaluated by two different methods: Pixel counting and Area overlapping (Cyto=Cytoplasm). ....................... 109
Table 5. 4– Comparison of the proposed segmentation method evaluated by the closed contour overlapping area and other studies (Nuc=nucleus, Cyto=cytoplasm, over-seg=over-segmentation, under-seg=under-segmentation)......................................................... 110

Table 5. 5– Confusion matrix for every classifier model representing the performance of every model classifying the training dataset................................................................. 119

Table 5. 6– Confusion matrix of the three models (SVM, KNN, and D.Tree) fusion and all models fusion representing the performance of every composite fused model classifying the training dataset. ................................................................. 120

Table 5. 7 – Comparison between the classifiers performance attributes on the training dataset with the fusion results using the majority-voting fusion method. The segmentation method used is the SVM based segmentation method......................... 120

Table 5. 8– The local minima of the search space values for the RBF sigma and the box constraint. ................................................................................................................................. 123

Table 5. 9– Classifiers performance parameters used to evaluate the classification process. ................................................................................................................................. 125

Table 5. 10– Comparison between the classifiers performance attributes with the fused results using the majority-voting fusion method. The segmentation method used is the SVM based segmentation method. ................................................................. 132

Table 5. 11– Comparison between the classifiers performance attributes with the fused results using the majority-voting fusion method. The segmentation method used is the ANN based segmentation method. ...................................................................................... 134
Table 5. 12– Comparison between the classifiers performance attributes with the fused results using the majority-voting fusion method. The segmentation method used is the watershed based segmentation method ............................................................. 135

Table 5. 13– The classifier ensembles with classification performance attributes and DST trust factor for the fused models .............................................................. 136

Table 5. 14– Correlation between manual classification and the proposed system classification output showing the resulting correlation (R) and significance (P) values for the two lymphocyte cell class (CLL and Normal) ............................................ 138

Table 5. 15– Decision correlation between the proposed system results and the flow cytometry device results for 11 CLL cases ................................................................. 139

Table 5. 16 – Comparison of the proposed CDDS and other CDSS for CLL detection. (CVM= cross-validation method) ............................................................................ 142
List of Figures

Figure 1. 1– The hierarchy block diagram of the proposed system showing the relationship between the research contributions, objectives, and methodologies. (MCS=Multiple Classifier system, SVM=Support Vector Machine, KNN= K-Nearest Neighbor, ANN= Artificial Neural Network, AdaBoost= Adaptive Boosting, DST= Dempster-Shafer Theory of evidence) .............................................................. 5

Figure 2. 1– General training and classification procedures for (n) classes supervised classification. ............................................................................................................. 23

Figure 2. 2– Multi-Layer Perceptron Artificial Neural Network with two hidden layers. ................................................................................................................................... 29

Figure 2. 3– The linear SVM classifier and the support vector. ........................................ 30

Figure 2. 4– Transforming the non-linearly separable dataset from the input space to the high dimensional space. ........................................................................................................ 32

Figure 2. 5– Linear combination of weak classifiers to form a stronger classifier model. ................................................................................................................................... 35

Figure 3. 1– Basic component of the proposed CDSS system for CLL cell classification. ................................................................................................................................... 48

Figure 3. 2– The proposed watershed based segmentation algorithm for normal and CLL lymphocyte cell, nucleus, and cytoplasm segmentation. ........................................ 52

Figure 3. 3– The proposed machine learning based segmentation algorithm for normal and CLL lymphocyte cell, nucleus, and cytoplasm segmentation using Support Vector Machine (SVM)/Artificial Neural Network and K-Means Algorithm with pixels’ color normalization. ................................................................................................................................. 56
Figure 3. 4– The segmented output masks used to extract features which are fed to the FSA to choose a subset of the features that have the highest separation power. 61

Figure 3. 5– General flow chart of a supervised classification process which includes training, testing (validating), and parameters tuning stages. 65

Figure 3. 6– Training process of the SVM: the data is loaded and divided into 10-folds then the cross-validation error is calculated and the optimum parameters are chosen. A test lymphocyte image is classified using an SVM model trained by the selected optimal parameters. 69

Figure 3. 7– The ANN training process. The training features are loaded into the ANN with the default parameters. The cross-validation error is computed and then the ANN topology is altered by increasing the neurons in the hidden layer and choosing the best topology based on the cross-validation error. 71

Figure 3. 8– Two classes K-NN classifier feature space. 73

Figure 3. 9– K-NN classification algorithm. The algorithm measures the features of the test lymphocytes and throws it into the features space along with the features of the training dataset then the nearest neighbor label is assigned to the test lymphocyte. 73

Figure 3. 10– Decision tree training flowchart. The re-substitution and cross-validation errors are used to determine the optimum number of leaf node. 75

Figure 3. 11– Binary decision tree classification process for test lymphocyte images. 76

Figure 3. 12– AdaBoost training and classification algorithm. 78

Figure 3. 13– The three different techniques used to create and train an ensemble of classifiers. 80
Figure 3. 14– The DST method used to calculate the trust factor of the fused classifier models. ................................. 83

Figure 4. 1– The proposed CDSS software architecture. ................................................................. 90

Figure 4. 2– The components of the controller module showing the data-centric architecture and the relationship between the internal components. ............. 92

Figure 4. 3– The proposed tailored wrapper design pattern. ......................................................... 96

Figure 4. 4– A sequence diagram showing the interaction between the client and the new algorithm to map the concrete interface of the new algorithm to the client interface. ......................................................................................................................... 97

Figure 5. 1– Sample images (CLL and normal) and the resulting output segmentation masks (a) Original images, (b) Extracted cell mask, (c) Manually segmented cell, (d) Extracted nucleus mask, (e) Manually segmented nucleus, (f) Extracted cytoplasm mask. ........................................................................................................................ 105

Figure 5. 2– Sample images (CLL and normal) and the resulting output with increasing the percentage suppression of the local minima (a) Sample CLL image (b) Sample normal lymphocyte image. ........................................................................................................ 106

Figure 5. 3– Sample images (CLL and normal) and the resulting output masks (a) Original images, (b) Extracted cell mask using an SVM, (c) Extracted cell mask using an ANN (d) Manually segmented cell, (e) Extracted nucleus mask using an SVM, (f) Extracted nucleus mask using an ANN (g) Manually segmented nucleus, (h) Extracted cytoplasm mask using an SVM, (i) Extracted cytoplasm mask using an ANN, (j) Cell output mask without pixel color normalization (the SVM method). ........................................................................................................ 108
Figure 5. 4– Nucleus under-segmentation error of the three proposed segmentation methods: the SVM, the ANN, and the watershed algorithm. ........................................ 112

Figure 5. 5– Cell accuracy estimation for 140 test images using the three proposed segmentation methods based on an SVM, an ANN, and a watershed algorithm. ...................................................................................................................................... 112

Figure 5. 6– The execution time of the three proposed segmentation algorithms for the 140 lymphocyte test images. ........................................................................................................................................ 113

Figure 5. 7– Distribution of the mean and standard deviation of the CLL and normal lymphocyte features. ........................................................................................................................................ 117

Figure 5. 8– Distribution of the mean and standard deviation of the selected CLL and normal lymphocytes features after removing the high mean and standard deviation features. ........................................................................................................................................ 117

Figure 5. 9– SBS wrapper external software component used to select the classifiers optimum features. ................................................................................................................................. 118

Figure 5. 10– SFS wrapper external software component used to select the classifiers optimum features. ................................................................................................................................. 118

Figure 5. 11– Receiver Operating Characteristics of the trained classifier models used by the proposed system with the SVM based segmentation algorithm. ................................. 121

Figure 5. 12– Output report generated by the proposed system for the CLL training images. .............................................................................................................................................. 122

Figure 5. 13– Output report generated by the proposed system for the Normal training images. .............................................................................................................................................. 122
Figure 5. 14– The SVM reinforcement learning repetition to choose the best parameters.
........................................................................................................................................ 124

Figure 5. 15– Receiver Operating Characteristics for the classifier models used by the proposed system with the SVM based segmentation algorithm. ......................... 125

Figure 5. 16– The ANN reinforcement learning repetition to choose the best number of neurons in a single hidden layer. ................................................................. 127

Figure 5. 17– KNN classifier cross-validation and re-substitution errors with increasing the number of neighbours (k). ................................................................. 128

Figure 5. 18– Full tree model fitted to the lymphocyte training dataset. ..................... 129

Figure 5. 19– The cross-validation and the re-substitution errors used to determine the number of leaf nodes for the decision tree classifier. ................................. 129

Figure 5. 20– The pruned decision tree model. .............................................................. 130

Figure 5. 21– The relationship between the test classification error and the cross-validation error with increasing the number of trees used by the AdaBoost algorithm, which is used to determine the optimum number of trees to be used. ................. 131

Figure 5. 22– The classification performance attributes for the different classifier models and the fused model results using the majority voting method. ......................... 132

Figure 5. 23 – Receiver Operating Characteristics for the classifier models used by the proposed system with the ANN based segmentation algorithm. ....................... 134

Figure 5. 24 – Receiver Operating Characteristics for the classifier models used by the proposed system with the watershed based segmentation algorithm. ............... 135
Figure 5. 25– Direct correlation between the manually pre-classified lymphocyte images and the classification of these images using the proposed system with the correlation R=0.66. .................................................................................................................... 137

Figure 5. 26– Sample output report generated by the proposed system. ....................... 140

Figure 6. 1– Touching lymphocytes and corresponding output (a) Original images, (b) Extracted cell mask, (c) Extracted nucleus mask, (d) Extracted cytoplasm mask. . 149

Figure 6. 2– The big picture of the “EzPatho” generic CDSS............................................. 150
### List of Symbols, Abbreviations, and Nomenclature

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdaBoost</td>
<td>Adaptive Boosting</td>
</tr>
<tr>
<td>ALL</td>
<td>Acute Lymphocytic Leukaemia</td>
</tr>
<tr>
<td>ANN</td>
<td>Artificial Neural Network</td>
</tr>
<tr>
<td>BBN</td>
<td>Bayesian Believe Network</td>
</tr>
<tr>
<td>CAD</td>
<td>Computer Aided Diagnosis</td>
</tr>
<tr>
<td>CCR</td>
<td>Correct Classification Rate</td>
</tr>
<tr>
<td>CDSS</td>
<td>Clinical Decision Support System</td>
</tr>
<tr>
<td>CGD</td>
<td>Conjugate Gradient Decent</td>
</tr>
<tr>
<td>CLL</td>
<td>Chronic Lymphocytic Leukaemia</td>
</tr>
<tr>
<td>CLS</td>
<td>Calgary Laboratory Services</td>
</tr>
<tr>
<td>CS</td>
<td>Client Server</td>
</tr>
<tr>
<td>DP</td>
<td>Decision Profile</td>
</tr>
<tr>
<td>DST</td>
<td>Dempster-Shafer Theory</td>
</tr>
<tr>
<td>FN</td>
<td>False Negative</td>
</tr>
<tr>
<td>FP</td>
<td>False Positive</td>
</tr>
<tr>
<td>FPR</td>
<td>False Positive Rate</td>
</tr>
<tr>
<td>FSA</td>
<td>Features Selection Algorithms</td>
</tr>
<tr>
<td>KB</td>
<td>Knowledge Base</td>
</tr>
<tr>
<td>KB-CDSS</td>
<td>Knowledge Base Clinical Decision Support System</td>
</tr>
<tr>
<td>KBK-SR</td>
<td>Kernel Bisecting K-means and Sample Removal</td>
</tr>
<tr>
<td>K-NN</td>
<td>K-Nearest Neighbour</td>
</tr>
<tr>
<td>LS</td>
<td>Layered System</td>
</tr>
<tr>
<td>MCS</td>
<td>Multiple Classifier System</td>
</tr>
<tr>
<td>MLA</td>
<td>Machine Learning Algorithm</td>
</tr>
<tr>
<td>MLP</td>
<td>Multi-Layer Perceptron</td>
</tr>
<tr>
<td>MPCNN</td>
<td>Multichannel Pulse-Coupled Neural Network</td>
</tr>
<tr>
<td>MVC</td>
<td>Module View Controller</td>
</tr>
<tr>
<td>NKB-CDSS</td>
<td>Non-Knowledge Base Clinical Decision Support System</td>
</tr>
<tr>
<td>OOP</td>
<td>Object Oriented Programming</td>
</tr>
<tr>
<td>PCA</td>
<td>Principle Component Analysis</td>
</tr>
<tr>
<td>PCNN</td>
<td>Pulse-Coupled Neural Network</td>
</tr>
</tbody>
</table>

xix
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDF</td>
<td>Probability Density Function</td>
</tr>
<tr>
<td>QP</td>
<td>Quad Programming</td>
</tr>
<tr>
<td>RBCs</td>
<td>Red Blood Cells</td>
</tr>
<tr>
<td>RBF</td>
<td>Radial Base Function</td>
</tr>
<tr>
<td>ROC</td>
<td>Receiver Operating Characteristics</td>
</tr>
<tr>
<td>SA</td>
<td>Software Architecture</td>
</tr>
<tr>
<td>SBS</td>
<td>Sequential Backward Selection</td>
</tr>
<tr>
<td>SD</td>
<td>Sequence Diagram</td>
</tr>
<tr>
<td>SFS</td>
<td>Sequential Forward Selection</td>
</tr>
<tr>
<td>SLL</td>
<td>Small Lymphocytic Lymphoma</td>
</tr>
<tr>
<td>SMO</td>
<td>Sequential Minimum Optimization</td>
</tr>
<tr>
<td>SOAP</td>
<td>Subjective Objective Assessment Plan</td>
</tr>
<tr>
<td>SOM</td>
<td>Self-Organized Map</td>
</tr>
<tr>
<td>SVM</td>
<td>Support Vector Machine</td>
</tr>
<tr>
<td>TN</td>
<td>True Negative</td>
</tr>
<tr>
<td>TNR</td>
<td>True Negative Rate</td>
</tr>
<tr>
<td>TP</td>
<td>True Positive</td>
</tr>
<tr>
<td>TPR</td>
<td>True Positive Rate</td>
</tr>
<tr>
<td>WBCs</td>
<td>White Blood Cells</td>
</tr>
<tr>
<td>WHO</td>
<td>World Health Organization</td>
</tr>
</tbody>
</table>
Chapter 1: Introduction

Chronic lymphocytic leukaemia (CLL) is a blood cancer which develops in the soft spongy centre of long bones known as bone marrow. It is characterized by the proliferation of abnormal lymphocytes in the bone marrow, which do not respond to cell growth inhibitors [1]. CLL has a widespread prevalence amongst adults in Canada [2][3]. Moreover, the CLL cells morphology and size variations may be similar to normal lymphocytes in the early stages [4] and it may require a pathologist to identify the lymphocyte as CLL. For these reasons the clinical course and the phenotypic presentation of CLL are highly diverse, and, there are limited treatment options, consequently [5]. Thus, the current clinical best practices emphasize delaying the treatment until a patient demonstrates either symptomatic or progressive disease, which do not necessarily correlate with the optimal treatment outcomes or long-term survival [5]. In fact there is a tendency for diagnostic pathology to rely on automated systems [6] to aid in diagnosis. There are, however, a few related works on automated CLL segmentation and classification (See Chapter 2 for a review). No single genetic mutation or abnormality responsible for CLL development has been identified. Rather, the disease is characterized by a variety of chromosomal abnormalities [102]. Unlike other types of leukaemia, there is no firm evidence linking environmental or occupational exposure with the incidence of CLL except for exposure to Agent Orange [103]. However, recent research has demonstrated an increased risk in first-degree relatives of patients with CLL. Therefore, a family history of CLL or other lymph proliferative disorder may also be a CLL risk factor [102].
Over the past several decades the applications of biomedical informatics for computer-aided diagnostics (CAD) and clinical decision support system (CDSS) have become essential in clinical settings and laboratory examinations. CDSS is defined as any computer program that can link health observations with health knowledge to influence health choices by clinicians for improved health care. Adaptations of CDSS powered by biomedical informatics in either complex or simple forms were seen as early as the 1970s [7]. Currently medical devices and high-throughput measurement systems produce thousands of images per patient in seconds, making it difficult for physicians to parse through the information to provide timely diagnoses and predictions. There is a significant need for development and improvement of CDSS in the medical laboratory.

The goal of the CDSS for CLL cell classification is to label the lymphocyte cells as CLL or normal cells, with a high trust factor which results from the fusion of multiple classifier systems (MCS). MCS is a special case of approaches that integrate several data-driven models (classifiers) for the same problem. A key goal is to obtain a better composite global model, with more accurate and reliable estimates or decisions. In the field of pattern recognition, fusion of multiple classifiers is currently used to solve difficult recognition tasks [52] and design high performance systems [67]. From a theoretical viewpoint, fusion of multiple classifiers allows the user to overcome the well-known limitations of the classical approach, which focuses on the search for the best *individual* classifier.

Software architecture (SA) is widely used to describe major design decision and high level design of software. SA represents the overall structure of a system in an abstract and structured manner. A good architectural representation scheme holds the key
to the effectiveness of SA description and usage. Data-centric SA is used in this research as it fits the nature of the CDSS. Many segmentation algorithms access the raw images and return segmented masks. Measurement algorithms process these masks and return a set of measured features. Features selection algorithms in turn select sub-features from the extracted features with minimum redundancy and maximum relevance. These selected features are the input to the MCS which returns a label for the input test image. The system results are offered to the hematopathologist in a report format which includes the classification results of the test lymphocyte images accompanied with a suggestion based on the percentage of the found CLL cells. The report includes a chart for the selected classifiers results and its majority voting fused results. The report contains a table showing the number of lymphocytes processed and the percentage of the found CLL and normal cells by the selected classifiers. The proposed CDSS system hosts a large amount of data, which is being accessed by many algorithms and go on to generate new data. This makes the data-centric software architecture the best fit for this type of system, which can accommodate the growth in the data size and the algorithms which use it.

The system uses a proposed tailored wrapper design pattern to integrate possible new algorithms into the system. The wrapper pattern can serve as a generic interface for any new added algorithms. The new algorithm interface is verified against built-in test procedures for different system components.

In the following sections of this chapter, I review research motivations, objectives, contributions, novelty and significance, and research methodology of this research; the last section outlines the structure of the thesis. Figure 1.1 shows the
1.1 Motivations

CLL is the common blood disorder (Leukaemia) widely spread amongst Canadian adults. CLL is the seventh deadliest cancer amongst Canadian adults. It has been estimated that for the year 2011, 5,000 new leukaemia diagnoses would be made in Canada. The Canadian Cancer Society’s estimate for leukaemia in Canada for 2012 is approximately 5,600 new cases. The 5-years survival rate is decreasing [2]. The relative 5-year survival rate estimated at 55% in 2012 whereas the survival rate was 80 % for men and 85% for women in 2009. The morphology and size of CLL cells are close to the normal lymphocyte [1]. This introduces difficulties identifying this disease in early stages. The problem with the current classification methods of the CLL is that they are dependent on the experience of the interpreting pathologist or hematopathologist and may remain silent for an extended period of time. There are few related works on this disease. Currently, there are a couple of commercialized systems for blood morphology diagnosis [8]. There is however, little work completed on the diagnosis of CLL based on automated analysis of microscopic blood images.
1.2 Research Objectives

The main objective of this research is to detect CLL through a flexible and adaptable CDSS and to achieve this objective the following sub-objectives are achieved:

- Speed up the processing time using fast algorithms. This will increase the throughput of the system resulting in a high screening throughput without sacrificing the classification accuracy. A trained hematopathologist can take 10 minutes for evaluation of 100 to 150 microscopic views.
• Accurate segmentation and classification. The application of the machine learning algorithms to accurately segment CLL cells overcomes the occlusion problem when the lymphocytes are tightly bound to the surrounding red blood cells (RBCs). The problem of over and under segmentation are no longer exist (see Chapter 3.3).

• Reusability, extendibility, and flexibility of the system.

The system can be used for:

1. **Research purposes**: The system can incorporate many segmentation and classification algorithms through its adaptive framework and tailored wrapper design pattern. The system can easily load a new algorithm and compare it against the current used algorithms. Performance in terms of processing time and accuracy can be calculated.

2. **Decision support tool**: The main purpose of the system is to classify the lymphocyte images and aid the decision of the interpreting pathologist.

3. **Microscopic images database**: many research communities develop their own database of images for testing and performance evaluation, such as computer science researchers who use Berkeley and MIT image databases [9]. Researchers conducting research in the field of the microscopic image analysis use images with different quality attributes when developing their algorithms. This makes it difficult comparing the performance of algorithms. For example: what is the image resolution? What are the characteristics of the imaging system (zooming factor, illumination, and specimens color staining) that might affect the processing steps?. Using the system to segment, classify, and annotate the lymphocyte
images with the image acquisition and quality attributes will provide a common ground to compare different segmentation and classification algorithms.

1.3 Research novelty and significance

The proposed system provides a tool for the hematopathologist to identify the CLL cells and supports his/her decision for the next step in the treatment process. The proposed system provides a user-friendly interface and flexible, adaptive easy-to-use framework for the user. A proposed interface adapter design pattern is included in the system to provide a generic interface to new added algorithms. The adapter gets the parameters from the new algorithm and provides it to the user to perform the parameter mapping for the input and output parameters between the system and the new algorithm. The existing CDSSs provide a concrete interface and it may not allow for the insertion of new algorithms.

The proposed system uses image processing algorithms such as watershed transform to segment the lymphocyte cell from the complicated blood smear image. The watershed algorithm suffers the variations of local minima due to the thick watershed lines which lead to over-segmentation and under-segmentation errors, where the cell may contain part of the background (over-segmentation) or lose part of the cell (under-segmentation). This segmentation problem is known as – occlusion problem– where the lymphocytes are tightly attached to the surrounding red blood cells (RBCs). Modification to the watershed algorithms and using of machine learning algorithms (MLA) such as the support vector machine (SVM) and the artificial neural network (ANN) are used to overcome these segmentation problems (See Chapter 3 for details).
Different classifier models have different structure, learning algorithms and separation capabilities [88]; and therefore different classifiers can make different errors when classifying the same features; however the classifiers aggregate result can provide a better accuracy for the features under test. An ensemble of classifiers is used to compose a composite classifier model [52], where the addition of a new classification algorithm should increase the classification performance of the composite model.

The proposed system uses a multiple classifier system (MCS) with a calculated trust factor based on the Dempster-Shafer theory of evidence (DST). The new added classification algorithm may increase the trust factor resulting in the addition of the new algorithm and a new composite classifier model is generated. If the trust factor stays the same or decreases after the fusion of the new added classifier, the system rejects this algorithm and informs the user.

By defining the common interface for the input and the output; the proposed system can be used for performance comparison of microscopic image segmentation and/or classification algorithms. The proposed system guides the user to hook a new algorithm into the adaptive framework. The proposed system checks the interface of the new algorithm and verifies its compatibility by testing it against built-in test procedures. The proposed system provides a reliable CDSS; as it incorporates three different segmentation algorithms ranked according to the segmentation accuracy and execution time performance. Furthermore a set of five different classifiers are used in the MCS structure. If any algorithm fails the system switches to the second best algorithm. This increases the reliability of the system and decreases the downtime. The proposed system
tests the existing algorithms by sending a known data input (i.e. image, features) to a specific algorithm and check its output.

1.4 Research Contributions

This research has the following contributions:

- Propose a new CDSS for CLL cell classification based on microscopic image analysis with MCS. The state of the art CellaVision™96 [8] cell classification equipment does not contain a software module to detect CLL cells.
- Propose 1% suppression of the watershed transform local minima as a pre-processing step of the watershed segmentation algorithm to control the problem of over and under-segmentation.
- The problem of over and under-segmentation has been significantly reduced and the occlusion problem no longer exist in the segmented masks when using the SVM and the ANN based segmentation methods.
- Design and implementation of the adaptive framework to host the proposed CDSS with a tailored wrapper design pattern, which provides an adaptive interface for hooking new algorithms easily into the framework.
- Quantify a trust factor for the MCS using the DST, which is used to judge the performance of the new composite classifier model resulting from adding a new classifier model to the existing composite classifier model.
1.5 Research Methodology

The peripheral blood smear images used in this research are pathologist pre-classified images of CLL, and normal lymphocyte cells. These images were obtained from the Department of Pathology and Laboratory Medicine, University of Calgary and Calgary Laboratory Services (CLS), Calgary, AB, Canada. Giemsa stained peripheral blood smear slides were used to acquire 6,345 images using the commercial state-of-the-art CellaVision™ DM96 system which is an automated image analysis system dedicated to locating and pre-classifying the various types of white blood cells (WBCs) in peripheral blood smears. The system also partially characterizes the red blood cell morphology and is able to perform platelet counts; however the system cannot detect the CLL cells and classifies them as normal lymphocytes.

Approximately 1010 out of 6,345 images were manually classified by human expert. The remaining images were sampled from positive CLL cases classified by a flow cytometry device. The segmentation results of the proposed system are validated using segmentation performance metrics for segmentation algorithms such as pixel counting and closed contour area overlapping. The classification results are validated against the significance test, correlation against a dataset of pre-classified images, and decision matching between flow cytometry results and the system classification results.

MATLAB ® image processing, image analysis and classification algorithms (the watershed, an optimal threshold, the SVM, the ANN, the K-means, etc.), and DST are used in the development of the proposed system [10].
1.6 Structure of the Thesis

In the following there is a brief description of the following chapters of the thesis. The thesis is structured as follows:

Chapter 2 contains background on the WBCs and CLL cancer, microscopic image segmentation, feature extraction/selection, WBCs classification, multiple classifier system and clinical decision support system. The software literature review is done on software architectures, software frameworks, and software design pattern. The background on the WBCs segmentation and classification consists of a review of basic definitions as well as the related works.

Chapter 3 presents the current shape of the system and methodologies used in its development. Details about the design of the algorithms used in the system are discussed in this chapter.

Chapter 4 presents the adaptive framework and software architecture of the proposed system. A detailed discussion about the data-centric architecture and the adaptive- framework with a tailored wrapper design pattern are presented in this chapter.

Chapter 5 presents detailed discussion of the results of every component used in the proposed system, overall performance and validation methods are discussed in details.

Chapter 6 is there to draw the final conclusions, limitations of the proposed system and the future works.
Chapter 2: Literature Review and Related Works

This chapter covers the background study for this research. Section 2.1 illustrates the background of the CLL and the CLL disease stages. Then, section 2.2 covers the CDSS. This is followed by the literature review and the related works of the basic components encompassing the CDSS, which are: image segmentation (section 2.3), features extraction, features selection and different algorithms used to rank the extracted features (section 2.4). Then this is followed by the literature review in classification algorithms and their related works to blood smear microscopic image classification (section 2.5 and section 2.6). Section 2.7 presents the literature review in software architecture (SA) and software framework that will be employed to host the proposed CDSS for CLL detection with adaptive software framework. Every section is concluded with an “In Summary” part, which communicates an in depth summary of the literature review and related works for that section. And finally a general conclusion, which focuses on the scope of this research: steps, methodologies, and algorithms used to develop the CDSS for CLL along with the host adaptive software framework, which are presented in section 2.8.

2.1 Literature Review : CLL and CLL staging

Microscopic examination of blood smear images is the main source of information that indicates changes in the development of specific diseases. Blood smear images consist of leukocyte cells (Eosinophil, Basophil, Neutrophil, Lymphocyte and Monocyte), red blood cells, platelets, and background.
CLL is a cancer of lymphocytes, which are blood cells involved in the body’s immune system. CLL/ small lymphocytic lymphoma (SLL) is classified by the World Health Organization (WHO) as a low-grade (slow-growing) non-Hodgkin lymphoma and is synonymous with SLL. The CLL cells are mainly found in the lymph nodes (glands), as in most other lymphomas. CLL is a disease which mainly affects older people and which has fairly limited treatment options. As more and more CLL cells accumulate, they can release chemicals which cause tiredness, weight loss and sweating. If they accumulate in the bone marrow they can also stop the bone marrow from working properly.

CLL is usually diagnosed by microscopic examination of blood smear films. It is suspected when the blood count shows a large number of lymphocytes. The microscopic examination of the CLL cells shows that CLL are small cells with condensed chromatin (which is found in the central nucleus of the cell) and very little cytoplasm. The diagnosis is confirmed by a technique called ‘immunophenotyping’, which involves the detection of the characteristic proteins (or ‘antigens’) on the surface of the lymphocytes.

After being examined, the CLL is described as being at a certain ‘stage’, which helps the physician decide when and how to treat the CLL. The staging system takes into account how many groups of lymph nodes are enlarged (most commonly the groups in the neck, the armpit or groin) and the blood counts.

Several staging systems have been proposed for (CLL), which identify three major subgroups and guide appropriate treatment decisions [11][12]. The modified Rai and the Binet systems are two widely accepted staging methods which are used in both patient care and for clinical trials. The modified Rai system is the most commonly used in
Canada. Table 2.1 shows the classification system for CLL as proposed by Binet [12]. Table 2.2 shows the Rai and Modified Rai classification staging system for CLL [11][12]. These staging systems are relatively simple, relying solely on physical examination and standard laboratory tests. However, the situation is more complex in patients diagnosed at an early stage of the disease, who account for up to 60% of all CLL patients. In this subset of patients none of the staging systems currently used can identify those patients who will have an indolent course and good prognosis as compared to those who will progress rapidly and finally die of their disease [12]. Several proposals have been made to identify criteria useful for the prognostic assessment of early CLL [11][13]. This is of particular importance considering that new prognostic factors, that in the future might assist identification of high-risk category of patients with early CLL [13].

Table 2.1– Binet classification system for CLL.

<table>
<thead>
<tr>
<th>Stage</th>
<th>Description</th>
<th>Median Survival (years)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Hemoglobin $\geq 100$ g/L and platelets $\geq 100 \times 10^9$/L and &lt;3 involved nodal areas</td>
<td>$&gt; 10$</td>
</tr>
<tr>
<td>B</td>
<td>Hemoglobin $\geq 100$ g/L and platelets $\geq 100 \times 10^9$/L and $\geq 3$ involved nodal areas</td>
<td>5</td>
</tr>
<tr>
<td>C</td>
<td>Hemoglobin $&lt; 100$ g/L and or platelets $&lt; 100 \times 10^9$/L and any number of involved nodal areas</td>
<td>2-4</td>
</tr>
</tbody>
</table>

Table 2.2– Rai and modified Rai classification staging system for CLL.

<table>
<thead>
<tr>
<th>Stage (Rai)</th>
<th>Description</th>
<th>Risk Status (Modified Rai)</th>
<th>Median Survival (years)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Lymphocytosis, with lymphoid cells $&gt;30%$ in the blood and/or bone marrow</td>
<td>Low</td>
<td>11.7</td>
</tr>
<tr>
<td>I</td>
<td>Stage 0 with enlarged node(s)</td>
<td>Intermediate</td>
<td>8.3</td>
</tr>
<tr>
<td>II</td>
<td>Stage 0–I with splenomegaly, hepatomegaly, or both</td>
<td>Intermediate</td>
<td>5.8</td>
</tr>
<tr>
<td>III</td>
<td>Stage 0–II with hemoglobin $&lt;110$ g/L</td>
<td>High</td>
<td>2-4</td>
</tr>
<tr>
<td>IV</td>
<td>Stage 0–III with platelets $&lt;100 \times 10^9$/L</td>
<td>High</td>
<td>2-4</td>
</tr>
</tbody>
</table>
In Summary: CLL is characterized by the progressive accumulation of functionally incompetent monoclonal lymphocytes. CLL/SLL is the most common adult leukemia in the Western world, accounting for approximately 7% of non-Hodgkin lymphomas [14]. In Canada, the median age at diagnosis is approximately 72 years, with 10% of cases diagnosed in patients younger than 50 years of age, with males representing approximately 56% of the cases. The five-year survival is approximately 80% for men and 85% for women [2]. In determining the optimal treatment for CLL, individual patient characteristics including performance status and disease stage must be considered.

2.2 Literature Review: Clinical Decision Support System (CDSS)

The automated systems for the early diagnosis of CLL are based on the morphological analysis of blood cells in microscopic specimen images [7]. Pathologists usually make a diagnosis by analyzing the morphologic and densitometry characteristics of specimen cells for distinguishing CLL from normal lymphocyte cells. The CDSS is defined as any computer system generating a partial or a comprehensive set of information used by the medical personnel as an aid in making medical decisions.

Clinical implementations of biomedical informatics methods in the form of computer-based decision support systems were seen as early as 1971, when Dombal’s system AAPhelp, developed at Leeds University which attempts to automate the diagnosis of acute abdominal pain [15]. In 1974 a system called INTERNIST-I [16], a rule-based expert system designed to aid the diagnosis of complex medical problems in internal medicine, was developed. A book by Greenes [17] outlines general concepts and future directions for the CDSS. Similarly, an article by Madabhushi [18] describes
development of computer-aided prognosis systems for predicting patient and disease outcomes using multi-scale and multimodal medical data. Miller’s article in 1994 provides a comprehensive list of important work conducted on diagnosis and decision support between 1954 and 1993 [7]. Similarly, a more recent article by Pearson provides a systematic review of computerized clinical decision support systems between 1990 and 2007 [19]. Tourassi discusses systems that provide diagnostic interpretations based on image texture analysis [20] and Stivaros [21] focuses on the impacts of decision support systems in clinical radiological practice.

A clinical decision support system known as Leuko has been developed for leukaemia diagnosis using a Naïve Bayes classifier [93]. The system is able to recognize the 5 basic types of normal WBCs and CLL. In this research a supervised Bayesian learning process segments blood smear images into four regions: nucleus, cytoplasm, erythrocytes, and plasma according to their color. The Leuko CDSS depends on the analysis of curvature [94], skeleton [95], and multi-scale fractal dimension [96]. The color space used for the analysis of neutrophil, eosinophil cytoplasm by the Leuko CDSS is the Hue-Saturation and Value, which requires more time for the differentiation of different types of leukocytes. This study has a CLL recognition accuracy of 72% and needs more samples to be considered and improvements to the problem of multimodal data representation of the color points that should include estimation of probability density function (PDF). Moreover features selection algorithm (FSA), may be used to select the most relevant features.

Research [97] investigates the use of multi-class Support Vector Machines (SVMs) classifier models to recognize the WBCs for future leukaemia diagnosis. The
results show the potential of SVMs to leukaemia diagnosis and indicate that a hierarchical tree-based multiclass strategy [98] can be better suited to a future update of the Leuko system. This study has a CLL recognition accuracy of 90% using 5 SVMs classifier models. The dataset used in study [93] are 151 CLL images, which are used by study [97] to design and validate the Leuko system. The method used to validate studies [93][97] is the Hold-Out method [66] which tends to be optimistic as there are a few number of images in the validation dataset, which may not represent the diversity in CLL cell morphology.

The proposed CDSS system is of a type known as non-knowledge-based (NKB-CDSS) and that is because it depends on the machine learning algorithms (MLA) to find some recognized CLL patterns in the images in a supervised manner. The other type of CDSS is the knowledge base (KB-CDSS) which includes a knowledge base, a communication tool, and inference rule engine. There have been several descriptions of CDSS and their characteristics [22]. Osheroff and colleagues have provided a detailed taxonomy of CDSS functions [23]. The CDSS encompasses a range of options, from general references, through specific guidelines for a given condition, to suggestions that take into account a patient’s unique clinical data. CDSS can include nationally recommended guidelines at one end of the continuum and customized order sets designed by an individual clinician.

The CDSS can potentially lower costs, improve efficiency, and reduce patient inconvenience. In fact, CDSS can sometimes address all three of these areas simultaneously, for example, by alerting clinicians to potentially duplicative testing. In general the aim of the CDSS is to assist, rather than to replace, the clinician [22]. The
CDSS may offer suggestions, but the clinician must filter the information, review the suggestions, and decide whether to take an action or what actions to take. Table 2.3 provides examples of CDSS that address a range of target areas.

Table 2.3 – Examples of CDSS interventions by target area of care.

<table>
<thead>
<tr>
<th>Target Area of Care</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preventive care</td>
<td>Immunization, screening, disease management guidelines for secondary prevention</td>
</tr>
<tr>
<td>Diagnosis</td>
<td>Suggestions for possible diagnoses that match a patient’s signs and symptoms</td>
</tr>
<tr>
<td>Planning or implementing treatment</td>
<td>Treatment guidelines for specific diagnoses, drug dosage recommendations, alerts for drug-drug interactions</td>
</tr>
<tr>
<td>Follow up management</td>
<td>Consequence orders, reminders for drug adverse event monitoring</td>
</tr>
<tr>
<td>Hospital, provider efficiency</td>
<td>Care plans to minimize length of stay, order sets</td>
</tr>
<tr>
<td>Cost reductions and improved patient convenience</td>
<td>Duplicate testing alerts, drug formulary guidelines</td>
</tr>
</tbody>
</table>

**In summary:** Computer-Aided Diagnostics (CAD) provide a second opinion to the user. It is created to assist the decision of image classification as a second opinion to the radiologist. In a CAD system the radiologist is the primary reader and decision maker. However, the CDSS assists the pathologist to make a primary decision on diagnosis. The CDSS has two types: knowledge base (KB) which has a precompiled data. The knowledge base contains the rules and relations of the compiled data which most often take the form of IF-THEN rules and deriving the decision rules used to extract the information behind the scene. The other type of the CDSS is the non-knowledge base (NKB-CDSS). It allows software to learn from previous cases and/or find patterns in clinical data. The scope of the proposed CDSS in this research is the development of a NKB-CDSS to detect CLL from blood smear microscopic images.
2.3 Literature Review: Microscopic Image Segmentation

Accurate segmentation of lymphocyte nucleus and cytoplasm from a microscopic blood smear image is a mandatory step to aid in the automatic detection and diagnosis of CLL. Segmentation is the process of correctly and accurately extracting different parts of an image. Leukocytes have different, wide variations of cell morphology and size which make it difficult to be segmented accurately. Many studies have addressed this problem. Multispectral white blood cell (WBC) segmentation using an SVM [24] showed robust, effective and insensitive results to blood smear staining and illumination condition. However the results showed low nucleus segmentation accuracy. This is due to the variation of the nucleus color. Research [25] obtained 98.9% maximum cell accuracy using feature scale-space filtering and watershed clustering. However it suffers from the over-segmentation and thick watershed lines due to plateau which can be reduced using smoothing algorithms. A framework for WBCs segmentation in research [26] shows accuracy of 92% for nucleus and 78% for cytoplasm using active contours and a Zack thresholding algorithm [27]. This low accuracy results from the utilization of the nucleus segmentation results into the Zack thresholding algorithm which further increases the error in the cytoplasm segmentation. Other research [28] has proposed a method for WBCs nucleus localization and segmentation using a combination of automatic contrast stretching, supported by image arithmetic operation, minimum filter and global threshold techniques. This proposed method is simple; however the results show that the proposed method manages to obtain a wide range of accuracy (85-98%). Color segmentation of acute leukemia images using HSI and RGB color spaces is proposed in research [29]. The results show that the method based on RGB color space did not give accurate results. An
online learning system for accurate cell segmentation is proposed in [30]. It simulates the visual attention of the human eyes. The results are promising however it requires a great deal of processing and it may not be suitable for limited resource systems. Over and under segmentation are problems arising when a segmented object contains parts from other objects or there are some missing parts of the segmented object. This is usually the case when using clustering algorithms such as the watershed algorithm for segmentation. Research [31] proposed an automatic white blood cell segmentation using stepwise merging rules and a gradient vector flow snake. It reduces the over-segmentation problem by 10.31% and the under-segmentation by 1.32%, but the algorithm is iterative and consumes a lot of system resources.

The SVM is widely used in many studies as it has many useful applications in many areas, such as pattern recognition, image processing, and bioinformatics. The majority of the studies of the SVM in hematopathology are dedicated for normal WBCs and acute lymphocytic leukemia (ALL) detection. Research [32] demonstrates an automated approach to clinical image segmentation using pathological modeling, a principal component analysis (PCA) and an SVM. Remarkable results have been achieved by applying the PCA to the extracted features and the results are used to train an SVM. However the algorithm is iterative. Many studies have addressed the problem of reducing the SVM training dataset to reduce the number of support vectors used in the classification process without sacrificing the performance. Most of the studies are based on the k-means clustering algorithm. Research [33] presents a new algorithm named kernel bisecting k-means and sample removal (KBK-SR) as a sampling preprocessing for the SVM training to improve the scalability. Other studies such as a chunking algorithm
[34], a decomposition algorithm [35], a sequential minimal optimization (SMO) [36], and an SVM light [37] are used for the SVM training. These algorithms tend to reduce the huge training task into a series of smaller sub-tasks in order to decrease the SVM training time. However the time complexity still needs further improvement in practice.

Localization and segmentation of lymphoblast cells using peripheral blood images is proposed in study [38], it gives an accuracy of 90-95% in restoring the lymphoblast pixels from the original image; and this is due to the color inconsistency. Color image segmentation using support vector machine and fuzzy C-means is proposed in [39]. It has the advantage of segmenting any type of images accurately and fast. However it suffers the problems of over and under-segmentation.

The Artificial Neural Network (ANN) has been used extensively during the past few decades. It has many applications in pattern classification as well as in image segmentation. Study [40] uses a Self-Organizing Map (SOM) neural network along with wavelets to segment the WBCs, which is a computationally expensive network. The results show that if the SOM training is performed on the wavelet-transformed image, it reduces the SOM training time and makes more compact segments. This method has the following advantages: it yields more homogeneous regions than those of other methods for color images, it reduces the spurious blobs, and it removes noisy spots. However the method is still computationally expensive. Many modified approaches to the classical ANN application have been developed in the literature to speed up the classification process of the ANN which may be used in both pattern classification, and image segmentation. Study [41] proposes a pulse-coupled neural network (PCNN) with multichannel (MPCNN) linking and feeding fields for color image segmentation. Pulse-
based radial basis function units are introduced into the model neurons of PCNN to
determine the fast links among neurons with respect to their spectral feature vectors and
spatial proximity. However the performance of the proposed method still needs
enhancements comparable to those of other popular image segmentation algorithms for
the segmentation of noisy images.

**In Summary**: The goal of image segmentation is the isolation of the regions of interest in
the image either by partitioning the image into connected semantic regions or by
extracting one or many specific objects from the image. The success of any classification
process is highly dependent on the segmentation results. Global image processing
techniques such as watershed, image arithmetic and the MLA such as an SVM and an
ANN with k-means clustering algorithm is used in the proposed CDSS presented in this
thesis to accurately extract the lymphocyte nucleus and lymphocyte cytoplasm for further
processing of CLL detection process. The success of the segmentation process is the key
factor in the success of the overall detection process.

### 2.4 Literature Review: Feature Extraction and Feature Selection

The complications of interpreting an accurate diagnostic decision in pathology are limited
by the lack of definitive measureable features for detecting and characterizing diseases,
and their corresponding histological features. The peripheral blood smear of patients is
routinely investigated for abnormalities; however, the delicate visible differences
exhibited by some disorders can lead to a significant number of false negatives during
microscopic examination of the peripheral blood smears. Figure 2.1 shows the general
block diagram of (n) classes supervised classification which involves training phase and testing phase to classify a test image.

Measuring object properties has been a subject of study since the early 1970s and is considered to be the culmination of considerable development [42][43]. It can also be used to discriminate between objects by measuring and comparing their properties. Feature extraction is the process of converting a given mask (cell, nucleus, and cytoplasm masks) into a set of measurements. There are many features that can be measured for a given object in an image [42]. There are three types of features, that can be measured for a lymphocyte cell:

1. Geometric features
2. Histogram based features
3. Intensity based features

Table 2.4 shows all the possible measurable features which can be measured for a lymphocyte cell. The feature selection is defined as choosing a subset of the extracted features that have minimum redundancy and maximum relevance to the object of interest. The generic purpose pursued is the improvement of the MLA classifier, either in terms of
learning speed, generalization capacity or simplicity of the representation. It is then possible to better understand the results obtained by the classifier, reduce storage, reduce the noise generated by irrelevant or redundant features and eliminate useless knowledge.

Table 2. 4– Measurable features of an object.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area</td>
<td>Scalar specifies the actual number of pixels in the region.</td>
</tr>
<tr>
<td>Perimeter</td>
<td>Scalar defines the distance around the boundary of the region of interest</td>
</tr>
<tr>
<td>Eccentricity</td>
<td>Scalar that specifies the eccentricity of the ellipse that has the same second-moments as the region of interest.</td>
</tr>
<tr>
<td>Equivalent Diameter</td>
<td>Scalar specifies the diameter of a circle with the same area as the region</td>
</tr>
<tr>
<td>Extent</td>
<td>Scalar that specifies the ratio of pixels in the region to pixels in the total bounding box.</td>
</tr>
<tr>
<td>Bounding Box</td>
<td>The smallest rectangle containing the region of interest</td>
</tr>
<tr>
<td>Convex Area</td>
<td>Scalar that specifies the number of pixels in Convex Image</td>
</tr>
<tr>
<td>Filled Area</td>
<td>Scalar specifying the number of on pixels in Filled Image.</td>
</tr>
<tr>
<td>Major Axis Length</td>
<td>Scalar specifying the length (in pixels) of the major axis of the ellipse that has the same normalized second central moments as the region of interest</td>
</tr>
<tr>
<td>Minor Axis Length</td>
<td>Scalar defines the length (in pixels) of the minor axis of the ellipse that has the same normalized second central moments as the region of interest</td>
</tr>
<tr>
<td>Solidity</td>
<td>Scalar specifying the proportion of the pixels in the convex hull that are also in the region.</td>
</tr>
<tr>
<td>Compactness</td>
<td>Scalar measures the efficiency of a contour to contain a given area</td>
</tr>
<tr>
<td>Mean</td>
<td>Scalar represents the average gray level value of the a given mask</td>
</tr>
<tr>
<td>Variance</td>
<td>Scalar represents a measure of inhomogeneity (second order moment)</td>
</tr>
<tr>
<td>Energy</td>
<td>Scalar represents the amounts of variation within a given mask</td>
</tr>
<tr>
<td>Entropy</td>
<td>Scalar represents the measure of non-uniformity in the image</td>
</tr>
<tr>
<td>Skewness</td>
<td>Scalar represents the symmetry of the PDF distribution of the gray level in a given mask</td>
</tr>
<tr>
<td>Kurtosis</td>
<td>Scalar represents the uniformity of the PDF distribution of the gray level in a given mask</td>
</tr>
</tbody>
</table>

The goal of the feature selection algorithm is to reduce the dimensionality of the classifiers input data by selecting the most distinctive features, which maximize the correct classification rate (CCR). There are basically two types of features selection
algorithm (FSA). The first type is the filter type, in which statistical analysis such as mutual information is used to rank the features according to the information represented by the features. The performance of a single feature classifier can be used to select features according to their individual predictive power. The predictive power of the feature can be measured in terms of error rate. Ranking criteria based on the CCR cannot distinguish between the top ranking variables where there are a large number of features that separate the data perfectly. The FSA based on filter type requires the use of the features PDFs which are not easily computed. However the probabilities can be estimated from the frequency counts in case of discrete features. Noise in data can affect the filter type as averaging of two redundant features can lead to a better CCR than a single noisy one.

The second type of the FSA is the wrapper type, which assess subsets of features according to their usefulness to a given classifier. The wrapper type [44] offers a simple and powerful way to address the problem of features selection, regardless of the chosen classifier algorithm. It is based on using the classifier performance to assess the relative usefulness of subsets of features. The wrapper type requires a methodology to search the space of all possible features subsets, which is computationally expensive.

There are two types of FSA search algorithm: sequential forward selection (SFS) and sequential backward selection (SBS). In SFS, features are progressively incorporated into larger subsets, whereas in SBS it starts with the set of all features and progressively eliminates the least promising ones [45].

**In summary:** Image analysis can provide several measures of an object’s structure by defining its properties in terms of area, perimeter, elongation, compactness, contrast, and
texture. Object measurements are normally computed from the binary representation of a segmented object or the gray-level intensity distribution within the object boundary. Object shape can be captured by measures of circularity, rectangularity, moments, and Euler number, among other features. Histogram measures capture the statistics of an object’s gray levels. The objectives of features selection are: improving the prediction performance of a classifier, providing faster and more cost-effective classifiers by reducing input data dimensionality, facilitating data visualization and data understanding, reducing the measurement and storage requirements, reducing training and utilization times, and providing a better understanding of the underlying morphology that generated the data.

There are two basic types for FSA: Filter type which selects subsets of variables as a pre-processing step, independently of the chosen classifier, and wrappers type, which utilizes the learning machine of interest as a black box to score subsets of variable according to their predictive power. Filters methods are faster than wrappers methods. However using linear classifiers with wrappers make it efficient, more accurate, and fast enough during training and classification.

2.5 Literature Review: Classification and Multiple Classifier System (MCS)

2.5.1 Artificial Neural Network (ANN)

The ANNs are powerful tools that can be trained to solve problems in a way similar to how the human brain works. It gathers knowledge by detecting the patterns and relationships in data and learns through experience. The ANN might consist of several thousand artificial neurons, and the output of one neuron becomes an input to another
neuron. There are several types of ANNs according to their structure and learning algorithms. The simplest neural network, called a perceptron, takes as input a real-valued vector of feature values, obtains a linear combination of them, and outputs a (1) if the combination is greater than a threshold and (−1) otherwise. This corresponds to the linear discriminant function:

\[ g(x) = w^T x - w_0 \]  

(2.1)

Where \( x \) is a feature vector, \( w \) is the vector of weights, and \( w_0 \) is the threshold.

Thus \( g(x) = 0 \) is the surface which separates items in class \( C_1 \): \( g(x) > 0 \) from items in class \( C_2 \): \( g(x) < 0 \), enabling the perceptron to act as a linear classifier for a two-class problem when the two classes are linearly separable. The perceptron must be trained, that is, the weight vector \( w \) is obtained, before it can be applied to classify an item without a class label. A simple approach to obtaining the weight vector is to start with random weights, apply the perceptron to classify a data item in the training set, and modify the weights whenever the item is misclassified. A gradient descent approach is used to find the weights which best separate the two classes in the training data [46] if the classes are not linearly separable. According to their structure the ANNs can be classified as feed-forward networks and recurrent networks [46]. In a feed-forward network, the neurons are generally grouped into layers. Signals flow from the input layer through the output layer via unidirectional connections. The neurons are connected from one layer to the next, but not within the same layer. In recurrent networks, the output of some neurons is fed back to the same neurons or to neurons in a preceding layer. The network topology which includes the number of units and their connectivity, is typically determined first, often by trial and error, prior to the start of training.
Each unit is a sigmoidal unit and it is a smoothly differentiable function:

\[ f(x) = \frac{1}{1+e^{-x}} \quad (2.2) \]

The commonly used learning algorithm for a multilayer network using sigmoidal units is the conjugate gradient decent (CGD) back-propagation algorithm. As in the case of a single perceptron, the error function is defined as the sum of the errors over all outputs units. The updates of the weight vectors are now more complex as there are multiple units as well as a layer of hidden units. The derivation of the weights, as well as various practical issues related to the implementation and application of the back-propagation algorithm, can be found in [46]. If the network has too few neurons, it may not be able to learn complex patterns; if it has too many neurons, it is likely to over-fit the data.

According to the ANN learning algorithm, the ANN can be classified to supervised learning, unsupervised learning, and reinforcement. In the supervised model the ANN requires the output in order to adjust its weight. In the unsupervised model, the ANN does not require the output, the ANN adapts purely in response to its input. The reinforcement learning algorithm employs a critic to evaluate the goodness of the neural network output corresponding to a given input [46]. Multi-layer perceptron (MLP) is a well-known type of ANN, which is usually used in classification problems. In MLP the neurons are grouped in many layers as shown in Figure 2.2. In this approach, during the training process of the network, the network compares its actual results with the desired output and then computes the error as shown in Eq. (2.3), which represents the mean square difference between the network output and the desired output. Through the back propagation algorithm the error will be presented many times to the input of the forward
activation place, and the process will continue until the actual outputs get closer to the desired output.

\[ \varepsilon(t) = \frac{1}{2} \sum_{i} (d_i(t) - y_i(t))^2 \]  

(2.3)

Figure 2.2– Multi-Layer Perceptron Artificial Neural Network with two hidden layers.

2.5.2 Support Vector Machine (SVM)

The SVMs are powerful tools for classification that can be considered as an alternative to the MLP. The SVMs were first introduced in 1992 [47]. The basic idea of the SVM is to find the linear classifier known as the hyper-plane that separates the classes. Figure 2.3 shows the linear SVM classifier and the support vectors used to define the separation margin of the classifier. In Figure 2.3 there is an ideal separating classifier – the hyper-plane – which increases the space between it and the nearest dataset points of different classes as much as possible. For separable classes as shown in Figure 2.3, an SVM classifier computes a decision function having a maximal margin ‘M’ with respect to the two classes.
Figure 2. The linear SVM classifier and the support vector.

There are two planes touching the boundary of dataset, \( w^T x + b = +1 \) and \( w^T x + b = -1 \); \( w \) is a vector perpendicular on the plane \( w^T x + b = +1 \). The maximum margin of the best classifier can expressed as \( M = \frac{2}{\sqrt{w^T w}} \). The decision boundaries can be found by solving the following constrained optimizing problem: Minimize \( \frac{1}{2} ||w||^2 \) subject to:

\[
y_i(w.x_i + b) - 1 \geq 0, \quad \forall_i
\]

(2.4)

where \( y_i \) is the hyper-plane equation and \( x_i \) is the data point set or basically the feature space. The Lagrange function formulation for this optimization problem is given by:

\[
L(w, b, \alpha) = \frac{1}{2} ||w||^2 - \sum_{i=1}^{n} \alpha_i (y_i(w.x_i + b) - 1), \quad \alpha_i \geq 0 \quad \forall_i
\]

(2.5)

By setting the derivative of the Lagrange function to zero:

\[
\frac{\partial}{\partial b} L(w, b, \alpha) = 0, \quad \frac{\partial}{\partial w} L(w, b, \alpha) = 0
\]

(2.6)

This yield:

\[
\sum_{i=1}^{n} \alpha_i y_i = 0, \quad \sum_{i=1}^{n} \alpha_i y_i x_i = 0
\]

(2.7)
The optimization problem can be expressed by:

$$\max_\alpha w = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i, x_j)$$  \hspace{1cm} (2.8)

where $x_i$ with the non-zero value of $\alpha_i$ are called support vectors.

In case of a non-linearly separable dataset as shown in Figure 2.4, the positive slack variable $\xi_i$ which controls the constrained condition in the hyper-plane equation is introduced leading to a soft margin classifier:

$$\min \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i$$  \hspace{1cm} (2.9)

$$\max_\alpha w = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i, x_j)$$  \hspace{1cm} (2.10)

where $C \geq \alpha_i \geq 0$

The parameter $C$ describes the trade-off between the maximal margin and the correct classification [48]. The parameter $C$ is chosen such that a larger $C$ corresponds to assign a higher penalty to errors. To solve non-linear classification problems, the LSVMs are applied to high dimensional spaces as shown in Figure 2.4, which transform the data into a high dimensional space. This means transforming from $(x_i, x_j)$ to $(\Phi(x_i), \Phi(x_j))$ which is known as kernel trick. This will lead to the following optimization problem:

$$\max_\alpha w = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j).$$  \hspace{1cm} (2.11)
Figure 2.4– Transforming the non-linearly separable dataset from the input space to the high dimensional space.

The common kernel functions are polynomial with degree (d), the Radial Base Function (RBF) with width (σ) and the sigmoid with parameter (k).

\[
\Phi(x_i, x_j) = ((x_i, x_j) + C)^d \\
\Phi(x_i, x_j) = e^{(-\frac{1}{2\sigma^2}\|x_i-x_j\|^2)} \\
\Phi(x_i, x_j) = \tanh(k(x_i, x_j) + \theta)
\]  (2.12) (2.13) (2.14)

2.5.3 Decision tree

Decision tree [49] is a popular technique used in classification problems as they are accurate, relatively simple to implement, produce a model that is easy to interpret and understand, and have built-in dimension reduction. A decision tree is a structure that is either a leaf, indicating a class, or a decision node that specifies some test to be carried out on a feature (or a combination of features), with a branch and sub-tree for each possible outcome of the test. The decision at each node of the tree is made to reveal the structure in the data. The traditional version of a decision tree algorithm creates tests at
each node that involve a single feature. As the tests at each node are very simple, it is easy for the domain expert to interpret the tree. There are several variants of oblique decision tree which differ in how the linear combination is obtained.

2.5.4 K-Nearest Neighborhood (K-NN)

The k-nearest neighbor algorithm is considered to be the simplest classifier model [50]. This algorithm belongs to the category of instance-based learning. In such techniques, the learning occurs only when the data items are to be classified. The classification algorithm typically classifies the data items as belonging to the nearest class that is represented by a set of measured features. The k-nearest neighborhood algorithm assigns to an unlabeled item the most frequently occurring class label among the k most similar data items. The similar data items are obtained using different distance metrics between the feature vectors such as Euclidean distance and city-block distance measurement metrics. The k-nearest neighbor can also be applied using weights, where the neighbors which are closer to the query item have larger weights.

2.5.5 Adaptive Boosting (AdaBoost)

Adaptive Boosting is founded on the notion of using a set of weak classifier models and pooling the classification results of such models to produce a stronger composite classifier model. In the sequence of weak models used, each classifier focuses its discriminatory power on the training samples misclassified by the previous weak classifier. The main reference for the AdaBoost algorithm is the original paper by Freund and Schapire [51]. AdaBoost maintains a probability distribution over all the training samples. This distribution is modified iteratively with each selection of a weak classifier. Initially, the probability distribution is uniform over the training samples. The weak
classifier, which is chosen at iteration $t$ of the AdaBoost algorithm is denoted $h_t$, and the
class label predicted by this weak classifier for the training data element $x_i$ is denoted $h_t(x_i)$. By comparing $h_t(x_i)$ with $y_i$ for $i = 1, 2, \ldots, m$, the error rate of the classifier $h_t$ can be assessed. The classification error rate of the weak classifier $h_t$ is denoted $\varepsilon_t$. The weak classifier $h_t$ is associated with $\alpha_t$ which denotes how much trust can be attained by this classifier. Obviously, the larger the value of $\varepsilon_t$ of a classifier model, the lower the trust level. The final classifier model is denoted as $H$. This classifier carries out a weighted aggregation of the classifications produced by the individual weak classifiers to predict the class label for a new data sample. The weak classifier can be a perceptron $a$ or a simple threshold.

The AdaBoost algorithm can utilize up to $(T)$ weak classifiers which can be as simple as individual attributes or, individual features that provide some discrimination between the objects of interest.

In the following steps the Adaboost algorithm is described for $t = 1, 2, \ldots, T$, classifiers:

1. For the probability distribution $D_t(i)$, use a weak classifier for the training data.
2. Apply the weak classifier $h_t$ as chosen in the previous step to all training data.
   \[ h_t: x \rightarrow \{-1,1\} \]  
   (2.15)
3. Estimate the classification error rate $\operatorname{Prob}\{h_t(x_i) \neq y_i\}$ for the $h_t$ classifier by
   \[ \varepsilon_t = \frac{1}{2} \sum_{i=1}^{m} D_t(x_i), |h_t(x_i) - y_i| \]  
   (2.16)
4. Calculate the trust factor for $h_t$ by
   \[ \alpha_t = \frac{1}{2} \ln \left( \frac{1-\varepsilon_t}{\varepsilon_t} \right) \]  
   (2.17)
5. Update the probability distribution over the training data for the next iteration:
\[ D_{t+1}(x_i) = \frac{d_t(x_i)e^{-\alpha_t y_i h_t(x_i)}}{z_t} \quad (2.18) \]

where the role of \( Z_t \) is to serve as a normalizer. This set a value for \( Z_t \) so that

\[ \sum_{i=1}^{n} D_{t+1}(x_i) = 1 \quad (2.19) \]

6. Repeat for \( T \) classifiers.

7. At the end of \( T \) iterations, construct the final classifier \( H \) as follows:

\[ H(x) = \text{sign} \left( \sum_{t=1}^{T} \alpha_t h_t(x) \right) \quad (2.20) \]

where \( x \) is the new data element whose class label need to be predicted on the strength of the information in the training data. If for a new data sample \( x \), \( H(x) \) turns out to be positive, the predicted class label for \( x \) is 1. Otherwise, it is (-1). Figure 2.5 shows the aggregation of the weak classifiers used by the AdaBoost algorithms to form a stronger classifier model which classifies non-linearly separable data.

![Diagram](image)

Figure 2.5– Linear combination of weak classifiers to form a stronger classifier model.

2.5.6 Multiple Classifier System (MCS)

An approach in classification which has gained much acceptance in the community of data mining and data fusion is the concept of ensembles or committees of classifiers,
which involves combining multiple models of classifiers to form a composite, more
stronger one. The idea behind this is very simple, in which the training dataset is used to
train several different models, each of which is used to assign a class label to a previously
unseen instance. These class labels are then combined suitably to generate a single class
label for the instance. This has been found to improve the accuracy of the resulting model
[52], however the process is computationally expensive and it is hard to understand how
the decision was obtained and this is depends on the fusion technique used which can be
one of the following techniques: majority-voting, maximum, minimum, average, sum,
decision templates and DST theory of evidence [53]. Ensembles have been used
extensively in the context of decision tree classification algorithms [54].

There are several ways to build classifier models from the same training dataset
[67]. However, the approaches vary in how they introduce randomization into the process
of model building so that different models are generated. One approach used in creating
ensembles is to change the instances which form the training set for each classifier in the
ensemble.

The most popular methods for this include [55]:

1. **Bagging**: In this approach, a new sample of the training set is obtained through
   bootstrapping with each instance weighted equally.

2. **Boosting**: In this case, a new sample of the training set is obtained using a
distribution based on previous results.

3. **Pasting**: In this approach, the ensemble of classifier models is grown using a
   subsample of the entire training set.
2.6 Related work: Using MLA and MCS to classify the WBCs

Reliable detection of pathological blood samples is of major importance in clinical laboratories [56]. Even though current automated cell counters used in hospitals are based largely on laser-light scatter principles, a quarter of the blood samples require microscopic review by experts. However, few algorithms [42][57] allow automatic cell classification using image processing. Study [58] applied two conventional classifiers to classify WBCs: a Bayes classifier and neural networks using four granulometric nuclei features without cytoplasm. An algorithm to optimize the pattern recognition of different white blood cell types in flow cytometry is introduced in study [59]. They used an SVM classifier to cluster parametric data in a multidimensional space. Research [57] presents an automated approach to a WBC classification method that uses a pairwise SVM classifier to label cytoplasm and nucleus features. Research [60] presents a two-phase methodology to analyze the morphology of abnormal leukocytes images for the classification of acute leukemia subtypes using image processing and data mining techniques.

There are several applications of the ANN which have been performed in the medical field. These include classification of blood cells [61] as well as diagnosis of lung [62] and ovarian [63] cancers. Research [64] has proposed the classification of blast cells by using 11 different types of leukemia. The classification result showed that an approximation of 70% of the leukemia samples have been classified correctly. Study [65] has proposed an automated leukemia detection that utilized fuzzy based blood image segmentation and an SVM to classify the lymphocytic cell nucleus as either lymphocyte
or lymphoblast. Fuzzy based two-stage color segmentation has been used to segment the WBCs. The final result showed that an accuracy of 93% has been achieved.

**In Summary:** The supervised classification starts with a collection of features, each of which has already been assigned to a class label. The goal is to build a “model” to distinguish amongst the different groups. The model can then be used to assign a class label to a test image which is described by a set of features, but does not have a class label. The labeled features are referred to as the “training” dataset as they are used to “train” the classifier to “learn” how to discriminate between unseen samples from different classes. Classification algorithms are usually evaluated by testing them on a part of the training data which has been set aside specifically for this purpose and not used during training – cross-validation –. The accuracy on this “test” is a measure of how well the classifier will perform on data it has not seen before.

Choosing what fraction of the data should be used for training and for validation is an open problem. Many researchers choose to use the leave-one-out cross-validation procedure; even though it is known to be a high variance estimator of generalization error [66] and give overly optimistic results, particularly when data are not properly independently and identically sampled from the “true” distribution. The leave-one-out procedure consists of removing one example from the training set, constructing the classifier on the basis only of the remaining training data, then testing on the removed sample. Another method of validation is the k-fold cross-validation in which the data is divided into (k) groups and only one group is used for validation, the remaining groups are used for classifier model training and the cross-validation is computed for this run. The whole procedure is repeated till all the groups have been used in the cross-validation
error calculation. At the end the average cross-validation error is computed and considered as the error expected when this classifier model is used to classify a new test sample.

Using of ensembles of classifiers provides several benefits, including a significant improvement in accuracy [67]. If the techniques involve using a subsample of either the instances or the features, creating (n) ensembles can take less time than creating (n) classifiers.

2.7 Literature Review: Software Architecture, Framework, and Design pattern

The concept of software architecture (SA) was first introduced in 1968 when layering was used in program development [68], then this concept was enhanced and structure of software was emphasized [69]. SA is responsible for incorporating quality in software by accommodating quality attributes and functional requirements. SA addresses the achievement of numerous quality attributes such as maintainability, reusability, expendability, and non-functional requirements (security, execution time, etc.) in a system [69]. SA is being widely used to describe a very high level design of large software systems. The main goal of a software architectural representation of a system is to identify the major components that constitute this system, and the interactions between these components, and represent them in a compact form [69].

Architectural styles can be divided into many types. The advantages and disadvantages of the SA style are listed below [69]:
1. Dataflow Systems: Pipes-and-Filters
This style has the advantage of supporting reuse and easy to maintain, however it has poor performance for interactive application.

2. Call and Return Systems: Data Abstraction and Object-Oriented Organization
This style has the advantage of concealing the implementation details which allow the object to be changed without affecting its clients. However, for an object to interact with another object it must know the identity of the other object.

This style has the advantage that complex problems may be partitioned into a series of steps. Ease of enhancement and reusability as each layer interacts with almost the layers below and above, changes to the function of one layer affect almost two other layers. However there is a difficulty in structuring some systems in a layered fashion impacting the performance of the system.

4. Independent Components: Event-Based, Implicit Invocation
This style has the advantage of invocation facilitates reuse, and make it easy to develop systems by allowing components to be replaced without affecting the interfaces of other components in the system. However the components abandon control over the computation performed by the system.

5. Network Based System: Client –Server (CS)
This style has the advantage of ease of maintainability, high level of security, ease of data update and maintain. However the high level of security and connection of many clients at the same time will overload the system and impact the performance.
6. Data-centric systems

This architecture style uses a central database to store all problem-related information. The single database provides convenient access to information, simplifying the process of extracting data in a variety of formats. This style has the following advantages:

1. Data Integrity: Data is entered once, at any time; erroneous, duplicated data is not possible.
2. Design reuse, accurate, and reliable data are available when needed.
3. View generation which alternates the views of the data.
4. Process flexibility in which the data management process is not constrained to application usage or sequence.
5. Data interaction is independent of the application: in which data can be accessed by the user through multiple applications.
6. Scalability in which the database can grow with application and domain needs.

The major drawback of this style is that its problem solving is only as good as the input data; errors in both the input data and the world knowledge can have a significant impact. SA can be compared according to software quality attributes defined by ISO/IEC 9126 standards [70]. Table 2.5 illustrate the common SA quality attributes and their definition.

Software frameworks have a potential to substantially improve software developer productivity and software quality. According to [71], the framework is defined as a set of classes that embodies an abstract design for solutions to a family of related problems. A framework can represent the core assets of a software product line. Software framework provides common codes as well as extension mechanisms for integrating user
implemented functions, termed hotspots [71], to handle domain variability. The software framework is based on a flexible and extensible structure which is tailored to some specific design requirements.

Table 2. 5—Common Software Architecture quality attributes and their definitions.

<table>
<thead>
<tr>
<th>SA quality attributes</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maintainability</td>
<td>The ease of a change to application architecture.</td>
</tr>
<tr>
<td>Reusability</td>
<td>The extent to which a program can be reused in other applications</td>
</tr>
<tr>
<td>Performance</td>
<td>The responsiveness of the system</td>
</tr>
<tr>
<td>Simplicity</td>
<td>Applying the principle of separation of concerns to the allocation of functionality within components</td>
</tr>
<tr>
<td>Scalability</td>
<td>The ability of the architecture to support large numbers of components, or interactions among components</td>
</tr>
<tr>
<td>Portability</td>
<td>Effort required transferring the program from one hardware and/or software system environment to another</td>
</tr>
<tr>
<td>Evolve-ability</td>
<td>The degree to which a component implementation can be changed without negatively impacting other components.</td>
</tr>
<tr>
<td>User Perceived Performance</td>
<td>The impact on the user in front of an application</td>
</tr>
<tr>
<td>Visibility</td>
<td>The ability of a component to monitor the interaction between two other components</td>
</tr>
<tr>
<td>Reliability</td>
<td>The expectation of a program to perform the required functions with the required precision.</td>
</tr>
<tr>
<td>Efficiency</td>
<td>The amount of computing resources and code required by a program to perform its function</td>
</tr>
</tbody>
</table>

A number of software frameworks have been developed to utilize patient data in an efficient way. One of the best known of these data entry frameworks is the SOAP format (subjective, objective, assessment, and plan) [72]. The SOAP format reflects how clinicians structure clinical information toward the ultimate purpose of solving patient problems, imbuing the data collection and analysis processes with detail, accuracy, scientific objectivity, and reproducibility [73]. The Patient-Centered Access to Secure
Systems Project framework (PCASSO) allows patient access to their medical records in a highly secured environment [74].

A software framework for the analysis of microscopic images is presented in [75]. This work highlights the software design considerations for microscopic image analysis. The authors argued that the proposed software framework may be used in conjunction with the other health care solutions to make certain types of analyses more efficient. Frameworks can be categorized as vertical or horizontal [69]. A horizontal framework provides system level services such as file access or device drivers. A vertical framework is more domains specific. It aims to provide abstractions of attributes and behaviours of a specific problem domain.

Christopher Alexander says, “Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice” [92]. Generally, a design pattern is a good and reusable solution to a common problem in software design. A single design pattern is usually described by well-defined attributes. The following are the attributes of the pattern:

- **Name:** is a term that concisely conveys the essence of the pattern.
- **Problem:** is a scenario which needs a solution and where the pattern is applicable.
- **Context:** is the case when the pattern can or should be applied.
- **Consequences:** are the results and trade-offs of applying the pattern, which may be conflicting forces that need to be resolved by the solution and is the main part of the pattern which instructs what to do.

The design patterns in software engineering are usually classified as follows [76]:
• Creational patterns deal with object creation mechanisms. They specify which object will create other objects. A creational pattern uses inheritance to vary the actual class.

• Structural patterns define how objects are composed to form larger structures.

• Behavioural patterns are concerned about the objects behaviour by assigning the responsibilities between them.

In addition to these three classes of patterns, which are related to object-oriented languages there is a fourth class of patterns which deals with multi-threaded programming [76].

Concurrency patterns are those types of patterns that deal with multi-threaded programming paradigm. They focus on tasks that execute concurrently, defining priorities and locks between them.

In summary: Software Architecture acts as a skeleton for the software development. SA needs to be created early during the software development and then the whole development process revolves around this skeleton SA is used to describe a high level design methodology of large software systems. SA represents the overall structure of a system in an abstract, structured manner. SA involves the description of the elements from which systems are built, interactions among those elements, patterns that guide their composition, and constraints on these patterns. Architectural style determines the vocabulary of components and connectors that can be used in instances of that style, together with a set of constrains on how they can be combined.

Software framework provides a solution to obtain extensible and reusable designs for a specific problem analysis. A framework aims to reuse designs as well as
implementation. It could have multiple implementations, but only one design. It should be possible to attach a new software module to the framework without any impact or modification or recompilation of the existing software. A software framework is a set of cooperating classes that make up a reusable design for a specific class of software.

2.8 Conclusion

CLL/SLL is a blood cancer. It accounts for 7% of non-Hodgkin lymphoma. It has a wide spread prevalence amongst Canadian adults with decreasing survival-rates. Despite being a common cancer there are, however, a small number of related works for automated CDSS for CLL cell classification.

The focus of this research is to develop a CDSS that is capable of classifying the CLL cells fast and accurately using data mining techniques (SVM, ANN, KNN, Decision tree, and Adaboost) with fusion mechanism that maximize the CCR.

The proposed system is based on the data-centric architecture with adaptive software framework. The chosen architecture and framework aim to facilitate hocking of new segmentation, feature extraction/selection, and classification algorithms. This dynamically expanding framework along with the tailored design pattern is the major contribution of this research.

Other contributions involve tuning the watershed algorithm for better segmentation results by introducing the concept of local minima suppression. The application of the SVM and the ANN as segmentation methods to segment the lymphocytes from the complicated images has totally eliminated the occlusion problem when the lymphocytes are being touched by the surrounding (RBCs). Also the over and
under segmentation problems have been significantly reduced. Using MCS enhances the performance of the CDSS. The state-of-the-art equipment CellaVision™96 cannot recognize CLL cells and label them as normal lymphocytes.

The validation of this study has been conducted using 6,345 lymphocyte images of CLL and normal cases. Approximately 1010 lymphocyte images were manually pre-classified, the remaining images (5335) were obtained from 11 positively identified CLL cases using a flow cytometry device which were blood sampled and analyzed using CellaVision™96. The images were fed to the proposed CDSS and decision correlation between the flow cytometry results and the CDSS outputs are drawn. The proposed system contributes in reducing the cost for CLL screening. This is true as the average cost of a flow cytometry device per patient is approximately $150 CAD; whereas the cost for the CellaVision™ 96 is approximately $5 CAD per patient [104].

In chapter 3 more details about the system components, methodologies, associated results, critical discussion of the results, and contributions are highlighted. Table 2.6 summarize the techniques used to develop the different components of the proposed system with the related thesis chapters.
Table 2.6 – The techniques used to develop the proposed system components with the reasons for using these techniques and the related thesis chapters.

<table>
<thead>
<tr>
<th>System Component</th>
<th>Techniques</th>
<th>Reason</th>
<th>Thesis chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segmentation</td>
<td>1-Watershed algorithm based segmentation</td>
<td>The watershed algorithm is a global image processing technique and requires no training. However it suffers from over/under segmentation problem which are overcome by using the SVM and the ANN algorithms.</td>
<td>Ch3.2, Ch3.3</td>
</tr>
<tr>
<td></td>
<td>2-ANN based segmentation</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3-SVM based segmentation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Segmentation</td>
<td>1-Mask pixels counting</td>
<td>Pixel counting is a fast technique to measure a segmentation algorithm accuracy, however it suffer the scattered pixel in a mask, which can be overcome by using closed contour area overlapping.</td>
<td>Ch3.4</td>
</tr>
<tr>
<td>accuracy measurement</td>
<td>2-Closed contour area overlapping</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature extraction/</td>
<td>1-SFS</td>
<td>The union of the results of these two methods are used to get a better feature representation of the CLL and Normal lymphocyte patterns.</td>
<td>Ch3.5</td>
</tr>
<tr>
<td>selection</td>
<td>2-SBS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Multiple Classifier</td>
<td>1-SVM</td>
<td>1- SVM maximize the margin between the classes.</td>
<td>Ch3.6, Ch3.7</td>
</tr>
<tr>
<td>System</td>
<td>2-ANN</td>
<td>2- ANN iterative optimization of linear classifiers.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3-KNN</td>
<td>3- KNN assign patterns to the majority class among K nearest neighbor using a performance optimised value for K.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4-Decision Tree</td>
<td>4- Decision Tree finds a set of thresholds for a pattern.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5-AdaBoost</td>
<td>5- Use a linear combination of classifiers</td>
<td></td>
</tr>
<tr>
<td>Classifier ensemble</td>
<td>Parallel ensemble training and majority-voting fusion method with Dempster-Shafer theory trust factor</td>
<td>The need for better classification accuracy and to quantify a variable which is capable of monitoring the accuracy after adding a new classifier model to the existing ensemble.</td>
<td>Ch3.8</td>
</tr>
<tr>
<td>fusion</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adaptive Framework</td>
<td>Creating adaptive framework for CDSS to detect CLL.</td>
<td>To provide a friendly interactive easy to use user interface</td>
<td>Ch4.1</td>
</tr>
<tr>
<td>Tailored design pattern</td>
<td>Modify the standard adaptor design pattern</td>
<td>To use and host a new algorithm into the proposed system</td>
<td>Ch4.2</td>
</tr>
</tbody>
</table>
Chapter 3: The Proposed System

This chapter presents the design of the proposed CDSS for CLL cell detection. The input to the proposed system is only lymphocyte images. The proposed system expects only pure lymphocytes at the input, and, thus, a normal WBCs classifier is used to classify the leukocyte images (Eosinophil, Basophil, Neutrophil, Monocyte, and Lymphocyte) at the input of the system which acts as a filter that allows only lymphocytes to be processed by the proposed system. The system results are offered to the hematopathologist in a report format which includes the classification results of the test lymphocyte images accompanied with a suggestion based on the percentage of the found CLL cells, which are one of the following: if the found CLL >= 70%, then the system suggests a CLL case; if the found CLL <40%, then the system suggests a normal case; otherwise a re-examination is suggested. The system block diagram is presented in Figure 3.1.

![System Block Diagram](image)

Figure 3.1 – Basic component of the proposed CDSS system for CLL cell classification.

Generally, the system is applicable to lymphocyte images which exhibit different image quality attributes. However images used to validate the system are high quality images acquired by the CellaVision™96 system [8] (See Section 3.1). The images are used to design, validate, and compare the performance of the proposed system to a flow cytometry device performance used to judge the cases. In this research the main image
quality considered is the contrast of the image, as images with high contrast will have defined boundaries for the cell, which can be used to differentiate the cell from the background. Poor contrast images will result in poor segmentation results and, thus, negatively impact the proposed system performance.

The system consists of four distinct components which are executed sequentially. The first component is the segmentation module, which is used to extract the lymphocyte cell from the complicated blood smear image and further segment it into nucleus and cytoplasm. Three different algorithms have been used to segment the lymphocyte cells into nucleus and cytoplasm masks, which are the watershed, the SVM, and the ANN algorithms. The algorithms' performance is compared to each other in terms of the segmentation accuracy and the execution time. The algorithms are ranked according to the comparison results. The algorithm with the highest accuracy and superior time performance has the highest rank and is selected to be the key segmentation algorithm in the proposed CDSS system. However, the main ranking criteria are the segmentation accuracy, in which an algorithm with higher accuracy and longer execution time can be used as the key segmentation algorithm.

The second component is the feature extraction and selection. A set of measurable parameters or features are measured for every segmented mask. These features are related to geometry, and statistical properties, and it can be used to generate invariant shape factor features. This is followed by selection of a subset of the features, which have the most discriminative power between the two classes (CLL and normal). The features selection step is important to remove the redundant features and speed-up the classification process.
The third component is the fusion of multiple classifier models which are used to classify a test lymphocyte image. In this step five classifiers of different structure and complexity are used in the system. The results of these classifiers are fused together to enhance the performance of the proposed system. The fusion method is the majority voting method, using the Dempster-Shafer theory of evidence [53] to calculate a trust factor of the composite classifier model generated from the fusion process.

At the end, the fourth component is a reporting tool to report the lymphocytes as CLL or normal. The report includes a chart and a table reflecting the number of cells found. The report includes a suggestion for further analysis of the patient images, and leaves the decision to the hematopathologist. The report includes a chart, which represents a histogram of the classified cells by each classifier, which represents a visual representation of the classifier performance for the user along with the fused model result. This could help increase the user confidence in making a treatment decision based on the system results. The percentage of CLL cells found is reported as a percentage of the total processed lymphocyte images per patient for the hematopathologist to aid his/her decision on the treatment process.

3.1 Lymphocyte Images

Giemsa stained peripheral blood smear slides were used to acquire 6,345 images using the commercial CellaVision™ DM96 system with a 100x oil-immersed objective. The system searches for the WBC and takes an image with the cell being at the center of the image. The images resolution is 363x360 pixels. The images acquired from the CellaVision™ DM96 are categorized as follows: 1010 images manually pre-classified
(CLL and normal) and 5335 images are acquired from 11 positively identified CLL cases using a commercial flow cytometry device. Table 3.1 shows the number of images used to design and validate the segmentation algorithms as well as the training and the validation images used by the system classifiers in the proposed system.

Table 3.1 – The images used to develop the proposed system.

<table>
<thead>
<tr>
<th>Number of images</th>
<th>Used for</th>
</tr>
</thead>
<tbody>
<tr>
<td>140 (CLL and Normal) pre classified lymphocyte images</td>
<td>Segmentation algorithms (Watershed, SVM, ANN).</td>
</tr>
<tr>
<td>129 CLL and 82 Normal pre classified lymphocyte images</td>
<td>Classification algorithms training.</td>
</tr>
<tr>
<td>662 CLL and 137 Normal pre classified lymphocyte images</td>
<td>Testing (validation) and learning reinforcement</td>
</tr>
<tr>
<td>5535 lymphocyte images from 11 CLL cases identified by a flow cytometry device</td>
<td>Decision correlation between the system output decision and the flow cytometry results.</td>
</tr>
</tbody>
</table>

3.2 Lymphocyte cell segmentation using a watershed algorithm and the Otsu’s threshold method

The purpose of the lymphocyte (CLL and normal) cell segmentation is to extract the lymphocyte nucleus and cytoplasm from other different parts in a microscopic blood smear image. Blood smear images consist of leukocyte cells “nucleus and cytoplasm”, red blood cells, platelets, and background. It is difficult to detect CLL disease early. This is due to the wide range of size and morphology of the lymphocyte cells, and the fact that CLL cell morphology and size are close to normal lymphocytes in the early stages of the disease [1]. The lymphocyte cell’s nucleus appears darker than the background, and the red blood cells. Moreover, large differences exist in the morphology and size of the
lymphocyte nucleus and cytoplasm. Figure 3.2 shows the proposed segmentation algorithm which is based on the watershed algorithm and Otsu’s thresholding method.

Figure 3.2– The proposed watershed based segmentation algorithm for normal and CLL lymphocyte cell, nucleus, and cytoplasm segmentation.
3.2.1 Nucleus Segmentation

Otsu’s thresholding approach is considered as a simple and powerful thresholding method for gray scale images [77]. It can be used to segment the nucleus depending on the fact that the nucleus appears darker than the surrounding components including the cell cytoplasm. The algorithm reads the color blood smear image, converts it to gray scale, computes Otsu’s threshold, then converts the image to binary image using the computed threshold value. The Otsu’s method is based on finding the threshold that minimizes the weighted variance within-class. The weighted within-class variance is given by:

\[ \sigma^2_{w}(t) = q_1(t)\sigma^2_1(t) + q_2(t)\sigma^2_2(t) \]  

(3.1)

where the classes’ probabilities are:

\[ q_1(t) = \sum_{i=1}^{t} P(i), \quad q_2(t) = \sum_{i=t+1}^{I} P(i) \]  

(3.2)

and the classes’ Mean are:

\[ \mu_1(t) = \sum_{i=1}^{t} \frac{iP(i)}{q_1(t)} \mu_2(t) = \sum_{i=t+1}^{I} \frac{iP(i)}{q_2(t)} \]  

(3.3)

Where the classes’ variance are:

\[ \sigma^2_1(t) = \sum_{i=1}^{t} [i - \mu_1(t)]^2 \frac{P(i)}{q_1(t)} \]  

(3.4)

\[ \sigma^2_2(t) = \sum_{i=t+1}^{I} [i - \mu_2(t)]^2 \frac{P(i)}{q_2(t)} \]  

(3.5)

The algorithm continues by applying the Canny edge detector on the thresholded image [78], then applying morphological dilation, hole-filling, and erosion operations to remove the small isolated pixels resulting from the thresholding and the Canny edge detection steps. Scattered isolated regions with a smaller area than the nucleus must be
removed to enhance the segmentation accuracy of the nucleus mask. This part of the algorithm creates a mask containing only the nucleus.

3.2.2 Cell Segmentation

For cell segmentation, the gray scale lymphocyte image is thresholded by Otsu’s method, followed by applying the Canny edge detector. It is necessary to remove the noise resulting from thresholding and edge detection by applying morphological dilation, hole-filling, and erosion operations. Distance transform is performed as a preprocessing step for the watershed algorithm [79]. The watershed algorithm is classified as a region based segmentation approach [80]. A local minima suppression of 1% is necessary before applying the watershed algorithm to reduce the effect of the over-segmentation and under-segmentation errors resulting from the watershed algorithm. This is due to the thick watershed lines resulting from converting the color image to gray scale image. The watershed transform considers the gray scale image as a composition of different surfaces, which have a certain distance to a common ground or local minima. Two different pixels having the same gray level belonging to two different objects will have the same watershed transform. The result of the watershed algorithm is a connected components matrix. The watershed distance transform for two or more occluded objects is one connected component, and an over-segmentation error exists in the results. A mask of the cell can be created by selecting the connected component at the center of the image. This is because the CellVision DM96™ system acquires the image with the lymphocyte cell being at the center of the image [8].
3.2.3 Cytoplasm Segmentation

The outputs of cell and nucleus segmentation are two masks containing only the cell and the nucleus of the lymphocyte. By simple pixel to pixel subtraction of these two masks; an accurate mask for the cytoplasm can be obtained as shown in Figure 3.2.

3.3 Lymphocyte cell segmentation using MLA and K-means algorithm

Machine learning algorithms such as the SVM and the ANN have a wide range of application in the pattern recognition field [59][61]. Other researches [39][57] apply the SVM and the ANN to segment the WBCs. Figure 3.3 shows the segmentation algorithm proposed in this research using the SVM and the ANN as machine learning algorithms. The algorithm is capable of segmenting the lymphocyte cells into nucleus and cytoplasm.

3.3.1 Nucleus Segmentation

In this section; the problem of lymphocyte nucleus segmentation is considered as a classification problem. The goal of the segmentation is to classify every pixel as nucleus pixel or background pixel. Therefore, it is important to select the training dataset for the nucleus region to robustly identify the nucleus’s pixels from the background pixels. The background here is defined as any non-nucleus’s pixels: “cytoplasm, RBCs, platelets and image background”. The algorithm for nucleus segmentation is shown in Figure 3.3. It starts with the training phase which includes 12 images for normal and CLL. The training data is extracted as follow: thresholding the original image using Otsu’s method [77] to select the nucleus pixels and removing the pixels that represent the color values of the cytoplasm trapped inside the nucleus region.
Figure 3.3 – The proposed machine learning based segmentation algorithm for normal and CLL lymphocyte cell, nucleus, and cytoplasm segmentation using Support Vector Machine (SVM)/Artificial Neural Network and K-Means Algorithm with pixels’ color normalization.
The process continues by collecting the pixels that belong to the nucleus only (non-zero pixels) and labeling it as –NUCLEUS–.

The algorithm creates, and captures a square mask of width 50 pixels at the training image top left corner to represent the background pixels, and label these pixels –BACKGROUND–.

Finally the algorithm combines both; the positive data –NUCLEUS– and negative data –BACKGROUND– into a feature matrix;(3xn) data points and (1xn) label column vector. Where (n) is the number of observations (pixels). This process is repeated for all the training dataset. The features matrix is composed of the color component (Red, Green, and Blue). The k-means clustering algorithm [81] is used to reduce the training time and the number of support vectors as well as the number of neurons in the hidden layer of the ANN topology. The k-means clustering algorithm divides the training data into 100 clusters which are represented by its centroid, and are experimentally chosen for best performance. Finally the training data is normalized and fed to the SVM and the ANN training algorithms. The training process is repeated many times to find out the best parameters for the SVM and the ANN algorithm such as the separation kernel and the solving method for the SVM and the number of neurons in the hidden layer of the ANN. The repetition of the training process aimed to get the best classification hits and minimum classification error. The color pixels of the unseen images are normalized and fed to the SVM and the ANN classifiers along with the support vectors produced from the training phase for the SVM and the neural net topology of the ANN to classify the color pixels into nucleus or non-nucleus pixel.
3.3.2 Cell Segmentation

The cell segmentation problem is also considered as a classification problem of the nucleus segmentation task. The only difference here is that the training dataset will include the cytoplasm color pixels plus the pixels belonging to the nucleus region. The algorithm for cell segmentation is shown Figure 3.3. It starts with reading the original image and the manually segmented mask of the lymphocyte cell. The subtraction of these two images gives a mask of the cell. The algorithm continues by collecting the pixels that belong to the cell only (non-zero pixels) and label it –CELL–.

The next step is to create a square mask of width 50 pixels at the image top left corner that represents the background pixels, and label these pixels as –BACKGROUND–.

The k-means algorithm is used to cluster the data points into 100 clusters, and these clusters are fed to the SVM and the ANN training algorithms. The SVM and the ANN training is repeated many times till the best performance of classification hit and minimum error are achieved. The unseen images are normalized and fed to the SVM and the ANN classifier along with the support vector resulting from the training phase of the SVM and the neural net topology of the ANN to classify the color pixels into cell or non-cell pixel.

It is worth noting that the basic differences between the SVM classification process and the ANN classification processes mentioned is the labeling of the training dataset. It is –NUCLEUS–, –CELL–, and –BACKGROUND– for the SVM training process, whereas it is 1, 1, and 0 for nucleus, cell, and background regions respectively for the ANN training process.
3.3.3 Cytoplasm Segmentation

The segmentation results are two masks containing only the cell and nucleus of the lymphocyte. An accurate mask for the cytoplasm can be easily extract by simple pixel to pixel subtraction of nucleus mask from the cell mask as shown in Figure 3.3. The cytoplasm region may be represented by very small number of scattered pixels, and in some images there is almost no cytoplasm exist in the lymphocyte cell. This is due to the wide variations of lymphocytes cell morphology.

3.4 Accuracy Measurement

To estimate the segmentation accuracy let $I_n$ and $I_c$ be the number of non-zero pixels in the output nucleus and cell masks. Let $I_{Mn}$ and $I_{Mc}$ be the number of pixels in the manually segmented nucleus and cell masks. The segmentation accuracy for the nucleus can be estimated by:

$$\text{Accuracy}_{\text{Nuc}} = [1 - \frac{I_{Mn} - I_n}{I_{Mn}}] * 100\%$$  \hspace{1cm} (3.6)

The segmentation accuracy of the cell can be estimated by:

$$\text{Accuracy}_{\text{cell}} = [1 - \frac{I_{Mc} - I_c}{I_{Mc}}] * 100\%$$  \hspace{1cm} (3.7)

Let $I_{cyto}$ be the number of non-zero pixels in the output cytoplasm mask. The segmentation accuracy for the cytoplasm can be estimated by:

$$\text{Accuracy}_{\text{cyto}} = [1 - \frac{(I_c - I_n) - I_{cyto}}{I_c - I_n}] * 100\%$$  \hspace{1cm} (3.8)

Closed contour area overlapping is another measurement metric used to evaluate the segmentation accuracy [82]. In which the overlapping area between the segmented mask and the ground truth image mask is computed. Higher segmentation accuracy
represented by higher overlapping area between the two masks. Let the intersection area between any ground truth and the corresponding output mask be $A_{Sec}$, the area of the ground truth image be $A_G$, and the area of the segmented mask be $A_M$. The overlapping area can be computed as follow [82]:

$$\text{Accuracy} = \left[ \frac{A_{Sec}}{A_G + A_M - A_{Sec}} \right] \times 100\% \quad (3.9)$$

The measurement of overlapping area can be modified to measure the over-segmentation and under-segmentation error as follow:

$$\text{Over-Segmentation} = \left[ 1 - \frac{A_{Sec}}{A_M} \right] \times 100\% \quad (3.10)$$

$$\text{Under-Segmentation} = \left[ 1 - \frac{A_{Sec}}{A_G} \right] \times 100\% \quad (3.11)$$

These formulas are used to measure the accuracy of the segmentation, under-segmentation, and over-segmentation for every output mask.

### 3.5 Feature extraction and feature selection algorithms

The output of the segmentation algorithm for every image is a set of three masks, which are cell mask, nucleus mask and cytoplasm mask as shown in Figure 3.4. These masks are used to extract descriptive and distinctive features that can easily differentiate between the CLL cells and normal lymphocyte cells. The measured features from the three masks are based on the geometrical shape, and the statistical measurements of the masks, which are used to extract scale, translation, and rotation invariant features.
Figure 3.4—The segmented output masks used to extract features which are fed to the FSA to choose a subset of the features that have the highest separation power.

A set of the measurable features initially used in this research are presented in Table 3.2, which includes the extracted features along with their physical interpretation. The range of values of these features must have a small variance between features; and, thus, every feature will be sensible. For example if a feature value span from 0.1 to 1 and another feature span from 10 to 100. The second feature will dominate and the first one will not be practical relative to second one.
Table 3. Features and features interpretation used in this study.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Feature interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eccentricity</td>
<td>F1: Circularity measure</td>
</tr>
<tr>
<td>Solidity</td>
<td>F2: Area/Convex Area</td>
</tr>
<tr>
<td>Compactness</td>
<td>F3: Efficiency of cell contour to contain cell area</td>
</tr>
<tr>
<td>Extent</td>
<td>F4: Area divided by the area of the bounding box</td>
</tr>
<tr>
<td>Mean, Variance,</td>
<td>F5-F10: Texture measurement of the cell</td>
</tr>
<tr>
<td>Variance, Energy,</td>
<td></td>
</tr>
<tr>
<td>Skewness, Kurtosis, Entropy</td>
<td></td>
</tr>
<tr>
<td>Nucleus Area/Cell Area</td>
<td>F11: Area/Convex Area</td>
</tr>
<tr>
<td>Eccentricity</td>
<td>F12: Nucleus area factor “scale invariant”</td>
</tr>
<tr>
<td>Extent</td>
<td>F13: Circularity measure</td>
</tr>
<tr>
<td>Mean, Variance,</td>
<td>F14: Area divided by the area of the bounding box</td>
</tr>
<tr>
<td>Variance, Energy,</td>
<td></td>
</tr>
<tr>
<td>Skewness, Kurtosis, Entropy</td>
<td>F15-F20: Texture measurement of the nucleus</td>
</tr>
<tr>
<td>Compactness</td>
<td>F21: Efficiency of nucleus contour to contain nucleus area</td>
</tr>
<tr>
<td>Extent</td>
<td>F22: Area divided by the area of the bounding box</td>
</tr>
<tr>
<td>Eccentricity</td>
<td>F23: Circularity measure</td>
</tr>
<tr>
<td>Solidity</td>
<td>F24: Area/Convex Area</td>
</tr>
<tr>
<td>Cytoplasm Area/</td>
<td>F25: Cytoplasm area factor “scale invariant”</td>
</tr>
<tr>
<td>Cell Area</td>
<td></td>
</tr>
<tr>
<td>Mean, Variance,</td>
<td>F26-F31: Texture measurement of the cytoplasm</td>
</tr>
<tr>
<td>Variance, Energy,</td>
<td></td>
</tr>
<tr>
<td>Skewness, Kurtosis, Entropy</td>
<td></td>
</tr>
</tbody>
</table>

For the sake of the supervised classification used in the proposed CDSS, a set of 129 CLL and 82 normal training lymphocyte images are used. There is no golden rule for the number of the training images used in a supervised classification. However the training images must reflect the common characteristics of both type of cells (CLL and normal). Images with redundant information must be removed from the training dataset.
also reactive cells or cells undergoing some reaction –mutation– should also be removed from the dataset.

The next step is to decide on the important features that have higher distinctive separation power between the two classes. This step is important to extract the features with minimum redundancy and maximum relevance, and reducing the dimensionality of the features matrix which helps in speeding up the classification process and boost the classification performance in terms of high CCR. There are two basic algorithms for features selection: filter type and wrapper type.

The filter type depends on ranking the features based on their discriminative power. This is achieved by using features correlation using statistical analysis such as paired t-test, entropy analysis, and mutual information analysis. Another ranking method is to rank the features according to their performance in one feature classifier. The filter type is fast, however it cannot rank two features that can completely discriminate between the two classes and dimension reduction may not be achieved.

The wrapper algorithm uses a specific classifier; usually Naïve Bayes network classifier, and sequentially adds or eliminates features to the classification process. Then the algorithm decides on the subset of data that give the highest CCR. However it is computationally expensive.

The algorithm used for features selection in this research is the wrapper method based on SFS and SBS. A software component [83] is used to analysis the features matrix and decides on the subset features with highest separation power between the two classes.
3.6 Multiple Classifier System

Supervised MLA takes a known input dataset with known classes, and aims to build a classifier model that produces a realistic forecasting in response to unseen lymphocyte test images. The proposed CDSS has five different classifiers which have different structure and complexity and therefore has different CCR.

This is the main idea of the classifier models ensemble fusion which depends on the fact that different classifiers make different errors. Therefore fusing the results of different classifiers have the effect of increasing the CCR of the whole classification process. The supervised classification process of any classifier consists of training, testing (validating), and parameters tuning stages. In the training stage the parameters of the classifier are initially chosen which represent the initial model’s parameters of the classifier. To determine the goodness of the chosen parameters a validation method is used to test the classifier performance [66].

The parameter of the classifier should be tuned for best performance to reduce the effect of over-fitting and increasing the CCR. After the tuning stage, the final classifier model is ready to classify a new lymphocyte test image. The flow chart of the supervised classification is shown in Figure 3.5.

Over-fitting generally occurs when a model is complex, such as having too many parameters relative to the number of measurements. A model which has been over-fit will generally have poor predictive performance, as it can exaggerate minor fluctuations in the data. In particular, a model is typically trained by maximizing its performance on some set of training data. However, its efficacy is determined not by its performance on the
training data but by its ability to perform well on unseen data. Over-fitting occurs when a model begins to memorize training data rather than learning to generalize from trend.

Several classifier performance validation methods are used in literature; such as: ‘Leave-One-Out’, ‘Hold-Out’, and ‘k-fold cross-validation’ [66]. Usually the k used in the k-fold cross-validation is 10, in which the cross-validation error could be calculated.
for 10 runs and the average error is used to represent the cross-validation error of the classifier to classify a new unseen test lymphocyte image. The cross-validation error is a better representation of a classifier ability to classify a new sample rather than the re-substitution error. The re-substitution error is the error resulting from classifying the training data and it is usually quite optimistic. A very small re-substitution error can reflect an over-fitting problem of the classifier, which means that if the classifier classifies a sample from the training images it will work well; however for new unseen lymphocyte images the CCR will be fairly small –low accuracy–. On the other hand a large re-substitution error reflects a poor training process and poor classifier model parameters. The method used for validation in this research is the 10-fold cross-validation method.

There are two types of classifier models the first model is based on the PDF of the features to predict a new sample and is known as parametric classifier model. Whereas the other model is known as non-parametric classifier model, in which the PDF cannot be estimated. In this research the classifier models are of the non-parametric type as the estimation of the PDFs for the different features were not possible to calculate and validate.

3.6.1 SVM classifier

The SVM model is a representation of the features dataset as points in the feature space, mapped in a way that the features of the CLL and normal cases are divided by a clear gap (Margin) which is as wide as possible (See Figure 2.3). New lymphocyte features are then mapped into the same feature space and is predicted to belong to a class (CLL or normal) based on which side of the gap they fall on. The features of the training dataset
of the lymphocyte images are overlapped which present a difficulty for an SVM classifier to linearly separate the two classes, however an SVM classifier can efficiently perform a non-linear classification using the kernel trick (See Figure 2.4), which maps the features into high dimensional feature space.

In this research the Soft Margin method is being used to determine the parameters of the hyper-plane that separate the two classes. The Soft Margin problem is solved by the Lagrange multiplier method [84]. The Soft Margin parameters are the slack variable $\xi$ which represents the degree of misclassification and C factor which represents the penalty cost of misclassification. Increasing C places more weight on the slack variable $\xi$, which means that the optimization attempts to make a stricter separation between classes. Equivalently, reducing C towards 0 makes the misclassification less important. Equations 2.9, 2.10, and 2.11 show the mathematical solution to the Soft Margin problem. The Soft Margin method chooses a hyper-plane that splits the features as cleanly as possible, while still maximizing the distance to the nearest cleanly split instances. The Soft Margin allows errors during training, which can allow the SVM to find solutions with noisy data and help prevent over-fitting and improve performance.

The quad programming (QP) [85] is used to solve the optimization problem for the SVM training which minimizes the L2-norm problem. Other optimization method can be used to solve the Soft Margin problem such as sequential minimum optimization (SMO) [36]. The SMO minimizes the L1 norm, which refers to using $\xi$ as a slack variable instead of their square. The SMO is a relatively fast algorithm compared to the QP which uses a good deal of memory, but solves the problem to a high degree of precision.
In this research the QP method and the Gaussian RBF are used to find the Soft Margin parameters as recommended by [86]. Figure 3.6 shows the method used to train an SVM and finding the best parameters for the hyper-plane. The method starts with loading the features dataset, and divides it to 10-folds for cross-validation process. The next step is to train an SVM with the default parameters settings which are [1,1] for the C factor and the sigma of the RBF used as the separation kernel. The cross-validation error is calculated for every training set of the features for 10 times and the average cross-validation error is reported. Then the method searches the parameters array for global minima that maximize the classification correct hits and minimize the over-fitting. Finally an SVM model is trained with the best optimized parameters for C and sigma. To classify a test lymphocyte image; the features of the test image are calculated and fed to the trained SVM classifier to label it as CLL or normal. The SVM package used in this research is the one available in MATLAB® R2011b.
Figure 3.6– Training process of the SVM: the data is loaded and divided into 10-folds then the cross-validation error is calculated and the optimum parameters are chosen. A test lymphocyte image is classified using an SVM model trained by the selected optimal parameters.

3.6.2 ANN classifier

The ANN architecture or topology plays an important role in the classification process, and the optimal topology depends upon the problem at hand. In this research the ANN topology is designed according to the number of dimensions of the features matrix: which are MLP network with 20 neurons in the input layer, 43 neurons in one hidden layer, and 2 neurons in the output layer to represent the classes. However, the output of the binary classifier is single valued and established by thresholding the results on the ANN. A
training of the ANN is conducted to select the number of neurons in the hidden layer based on the cross-validation error. The training was stopped when the cross-validation error started to increase in order to avoid network over-fitting. This stopping condition differed from network-to-network topology. Therefore, the exact moment in which the training stopped depends on the training set.

The training process is meant for determining the optimum weights and biases that produce the best ANN topology. Prior to the training, all network weights were initialized with random values in the range [-1, +1]. The ANN is trained with the conjugate gradient descent (CGD) back-propagation algorithm [46]. The back-propagation training method is simple even for complex models having hundreds or thousands of features. The ANNs are thus a flexible heuristic technique for conducting statistical pattern recognition with complicated models. Figure 3.7 shows the method used to train an ANN and finding the number of neurons in the hidden layer as well as classifying a test lymphocyte image. The method starts with loading the features dataset, and divides it to 10-fold for cross-validation process. The next step is to train an ANN with the default weights settings which are random and with 5 neurons in a single hidden layer. The cross-validation error is calculated for every training set of the features for 10 times and the average cross-validation error is reported and stored in an array. To determine the best number of neurons in the hidden layer; the algorithm increases the hidden layer neurons number by one and then calculates the cross-validation error again. At the end, the error array will have 95 values correspond to increasing the number of neurons in the hidden layer. The best topology is chosen based on the minimum cross-validation error for both classes. To classify a test lymphocyte image the features of the
test image are calculated and fed to the trained ANN classifier with the optimum number of neurons in a single hidden layer to label it as CLL or normal.

Figure 3.7– The ANN training process. The training features are loaded into the ANN with the default parameters. The cross-validation error is computed and then the ANN topology is altered by increasing the neurons in the hidden layer and choosing the best topology based on the cross-validation error.
3.6.3 **K-NN classifier**

The K-NN classifier is considered as a non-parametric classifier model as it does not require the features PDF. It is known as instance-based learning as all computation is deferred until classification. K-NN classifier classifies a sample lymphocyte test image by assigning it the label (CLL or normal) of the most frequently represented among the k nearest samples.

The nearest-neighborhood classifier relies on a metric or “distance” function between features which depends on the features extracted from the training data directly. The features of the unseen test lymphocyte image are extracted and thrown into the features space along with the features of the training dataset then the algorithm measures the distance of the k neighbors and check for majority vote. The K-NN classifier used in this research is based on 1-NN classifier. An illustration of a two-class K-NN classifier is shown in Figure 3.8. The flowchart of the algorithm used to classify the lymphocyte images using K-NN classifier is shown in Figure 3.9. The parameters of the K-NN classifier are picked up based on the cross-validation and re-substitution errors.

The algorithm starts by loading the training data and reading cell, nucleus, and cytoplasm masks of the unknown test lymphocyte image. The selected features by the FSA are computed. The K-NN algorithm computes the Euclidean distance between these measured features and the training dataset features. The algorithm continues by rank-order the instances in the training dataset features in ascending order according to the Euclidean distance. Therefore the class of the instance that is with the smallest distance to the unknown lymphocyte image is chosen by the algorithm to be the class of the unknown lymphocyte.
Figure 3.8– Two classes K-NN classifier feature space.

Figure 3.9– K-NN classification algorithm. The algorithm measures the features of the test lymphocytes and throws it into the features space along with the features of the training dataset then the nearest neighbor label is assigned to the test lymphocyte.
3.6.4 Decision tree

During the classification process using a decision tree, a test lymphocyte features is presented to the top decision node, and depending upon the answer (less than, equal to, or greater than), the decision flow passes to the right or to the left through the tree to a node at the next level. This process continues until the features come to a category label (CLL or normal). In a decision tree the links must be mutually distinct and exhaustive; that is, one and only one link will be followed, and, thus the decision can be easily interpreted for any particular test lymphocyte features as a combination of decisions along the path to its corresponding leaf node.

Decision tree is quite complicated and must be pruned to aid interpretation. Each decision outcome at a node is called a split, because it corresponds to splitting a subset of the training dataset. The root node splits the full training dataset; each successive decision splits a proper subset of the data. In general, the number of splits is set by the designer and could vary throughout the tree. If the tree continues to grow fully until each leaf node corresponds to the lowest level (label), then the data have typically been over-fit, however if the splitting is stopped too early, then the error on the training data (re-substitution error) is large and hence performance maybe degraded. The number of splits which control the number of leaf nodes is determined using the cross-validation and re-substitution errors. The design process of the decision tree focuses on deciding which feature test should be performed at each node. The popular measure that can be used to decide on the features to be used at each node is the entropy measurement [87] of a set of features at a specified node, and it calculated as:

\[ I(N) = - \sum p(w) \log p(w) \] (3.12)
where \( I \) is the entropy of a set of features \( w \) at node \( N \) with probability \( p(w) \).

The algorithm used to train and tune the decision tree parameters is illustrated in Figure 3.10. The training phase starts by loading the training data and divides it into training and test data for cross-validation. This is done by using \( k \)-fold cross-validation of (10) divisions of the training data. This is followed by fitting a classification tree of default parameters to the training data. Then the algorithm computes the re-substitution and cross-validation errors with varying the terminal leaf node count. From this, an optimal leafs for the tree can be concluded. This is when the re-substitution and the cross-validation error are close. The last step is to prune the tree to this optimal level.

Figure 3.10– Decision tree training flowchart. The re-substitution and cross-validation errors are used to determine the optimum number of leaf node.
Figure 3.11 shows the algorithm used to classify a test lymphocyte image using a binary classification tree. The algorithm starts by reading the cell, the nucleus, and the cytoplasm masks of the unknown test lymphocyte image. Then it creates a classification tree with 3 leaves (See Figure 5.16). The classification tree is used to predict the class label of the test image.

3.6.5 Adaptive Boosting

Boosting is a general method which attempts to enhance the accuracy of any given learning algorithm. Freund and Schapire, proposed in 1996, the Adaptive Boosting (AdaBoost) algorithm [51]. The idea behind adaptive boosting is to weight the data instead of randomly sampling it and discarding it. The training data for the weak learners (weak classifier models) is considered according to some weights. Initially equal weights are assigned to each training pattern. For subsequent iterations the classification error in the previous iteration is used to update the weights for the next iteration. The update of
the weights is such that the weights of incorrectly classified examples are increased so that the weak learner is forced to focus on the hard instances in the training dataset. It has been shown empirically that AdaBoost with decision trees has excellent performance, being considered the best off-the-shelf classification algorithm [55]. AdaBoost is resistant to over-fitting despite the fact that it can produce combinations involving very large numbers of classifiers. As the number of iterations increases, the processing time becomes significant. The weak learners used by the AdaBoost algorithm in this research are binary decision trees. The tree parameters are the same as the one derived in Section 3.5. Figure 3.12 shows the algorithm used to train an AdaBoost, finding the number of iterations – the number binary decision trees – required to classify unseen test image. The algorithm starts with loading the features dataset, and divides it to 5-fold for cross-validation process. Then it creates an ensemble of 500 binary decision trees. For the sake of fast training the number of k-folds is decreased from 10 to 5 and the cross-validation error is calculated for only 5 times. The algorithm continues by increasing the number of trees and calculates the cross-validation error. The number of trees in the ensemble can be found by choosing the number of trees beyond which the cross-validation error starts to increase. Then the ensemble is trained and combined together as outlined in Equations 2.15, 2.16, 2.17, 2.18, 2.19, and 2.20 (see section 2.5.5).

To classify a test lymphocyte image, the features of the test image are calculated and fed to the combined final classifier to label it as CLL or normal.
3.7 Reinforcement Learning

The most typical way to train a classifier is to present an input, compute its tentative category label, and use the known target category label to improve the classifier performance through classifier model parameters adjustments. This processed is known as reinforcement learning or learning with a critic, in which no desired category signal is
given; and the only teaching feedback is that the tentative category is right or wrong. In pattern classification, it is most common that such reinforcement is binary either the tentative decision is correct or it is not. A training dataset of lymphocyte images which were previously labeled by an expert hematopathologist are used in the reinforcement learning process for the proposed system (See Table 3.1).

The SVM, the ANN, the KNN, the decision tree, and the Adaboost classifiers are used to classify these images, and the best settings that maximize the CCR are chosen for every classifier model.

### 3.8 Multiple classifiers fusion

Multiple classifiers fusion is a method used to enhance the classification results in which a bundle of different classifier models, which are trained on a part or the complete training dataset; are combined together to come up with a better decision for the test sample. There are three techniques used to train a bundle of classifier models – ensemble – which are parallel, serial, and hybrid, as illustrated in Figure 3.13.

The best performance is achieved by combining both, different features dataset and different classifiers trained on the same features dataset. In this research the classifier ensembles methodology is based on combining a group of classifier models trained on the same features dataset [54].
The idea of building ensembles of classifiers has gained interest in the last decade [54]. Instead of building a single complex classifier model it may be better to design a combination of several simple classifier models. In this research five classifiers of different structures are trained on the same dataset and the classification results of every individual classifier are combined in order to produce the final decision.

There are many methods used in the literature to fuse the results of a classifier ensemble [54]. In this research the majority voting is used to fuse the classifiers results and the DST is used to calculate the trust factor of the ensemble. Assuming that the individual classifiers are uncorrelated, the majority voting of an ensemble of classifiers should lead to better results than using individual classifier. To fuse the results of the five classifiers based on the majority voting; the classifiers results for a given test image are summed up and thresholded, and if the sum is equal to or greater than (three), then the final decision for the cell is CLL otherwise it is a normal cell.
The DST can handle different types of uncertainty, particularly the uncertainty due to the lack of knowledge and it has been widely used in a variety of academic and industry fields due to its obvious advantages [53]. Unlike the traditional probability theory, the sum of the belief in a fact and its negation need not to be (one). Similarly the belief values can be associated with not only individual facts, but also with sets of facts. DST characterizes the subjective uncertainty well without the assumptions of the traditional probability theory. DST allows the expert to only give the degree of belief that one is confident with when there is a little information to evaluate the probability, thus, in the extreme situation, the value can be 0, which indicates that no information is available to make a judgment [54].

Decision profile matrix (DP) [54], is a representation which allows presenting the combination rules from a unified perspective, in which DP (x), for an instance x, consists of elements $d_{t,j} \in [0, 1]$, which represent the support given by the $t^{th}$ classifier to class $\omega_j$. The rows of DP(x), therefore, represent the support given by individual classifiers to each of the classes, whereas the columns represent the support received by a particular class from all classifiers. The decision template formulation becomes useful in describing DST as an ensemble combination rule: let $DT_j^t$ denote the $t^{th}$ row of the decision template $DT_j$, and $C_t(x)$ denote the output of the $t^{th}$ classifier, that is, the $t^{th}$ row of the decision profileDP(x): $C_t(x) = [d_{t,1}(x), \ldots, d_{t,C(x)}]$. Instead of similarity, proximity $\Phi_{j,t}(x)$ of the $t^{th}$ classifier’s class $j$ decision template $DT_j^t$ to this classifier’s decision on instance $x$, $C_t(x)$ is calculated [54]

$$DT_j = \frac{1}{N_j} \sum_{x \in X_{j\omega_j}} DP(X_j)$$

(3.13)
where the differences (calculated as distances in Euclidean norm) in both numerator and denominator are converted to similarities representing proximities using the reciprocal operation. The denominator is really a normalization term, representing the total proximity of $t^{th}$ classifier decision to the decision templates of all classes. Based on these proximities, belief or evidence is calculated so that the $t^{th}$ classifier $C_t$ is correctly identifying instance $x$ into class $\omega_j$,

$$ b_j(C_t(x)) = \frac{\phi_{j,t}(x) \prod_{k \neq j}(1 - \phi_{k,t}(x))}{1 - \phi_{j,t}(x) \prod_{k \neq j}(1 - \phi_{k,t}(x))} $$

(3.15)

Once the belief values are obtained for each source (classifier), they can be combined by the DST rule of combination, which simply states that the evidences (belief values) from each source should be multiplied to obtain the final support for each class:

$$ \mu_j(x) = K \prod_{t=1}^{T} b_j(C_t(x)) $$

(3.16)

where $K$ is a normalization constant ensuring that the total supports for $\omega_j$ from all classifiers is 1. Figure 3.14 show the application of the DST in combining the 5 classifiers output to estimate the fused decision for the lymphocyte test image.

In this research the DST is used to calculate a trust factor for the different combinations of the used classifier models. The combinations are all the possible aggregates of a group of 3 and 5 classifier models from the used five classifier models. The can use the aggregate of any number of classifiers; however the proposed system tends to mimics the human consultation situation, which usually based on 3 or 5 experts to come up with a decision. For every aggregate the DST is used to calculate a trust factor of this aggregate, and the aggregate with the highest trust factor, smallest false positive
rate (FPR), and highest accuracy is used by the system to classify a test lymphocyte image. In this way the system can decide on the best aggregate and a new added classification algorithm can be judged for its contribution to the classification process, in which the contribution is considered significant if the trust factor of the new aggregate is increased and thus the system accepts the new algorithm otherwise it declines the addition of the new algorithm. The new aggregate trust factor will remain the same if the new added algorithm classification result is exactly the same as one member of the current composite classifier model.

![Diagram](image)

**Figure 3.14**—The DST method used to calculate the trust factor of the fused classifier models.

### 3.9 System output

The output of the proposed system is presented in a report format, in which a comparison chart is presented for the chosen classifier models including the percentage of CLL and normal cells found by each individual classifier model and the fusion of the SVM, KNN, and Decision tree classifiers. The number of CLL cells found is reported as a percentage
of the total processed lymphocyte images per patient for the hematopathologist along with a suggestion to aid his/her decision on diagnosis.

3.10 Conclusion

In the proposed CDSS, the lymphocyte image is first segmented to extract the lymphocyte cell, nucleus, and cytoplasm from the complicated background. Distinctive features are then extracted and manipulated to remove the redundant ones. The selected features are used to train a group (5 classifiers) of classifier models. The trained models are used to label unknown test lymphocyte images. Then the system fuses the results of the SVM, KNN, and Decision tree classifiers to composite a more reliable classifier model and the results are presented to the user in a report format.

Segmentation is a crucial step in the process of CLL classification. Three algorithms are used to segment the lymphocyte cell, and furthermore segment it into nucleus and cytoplasm. The algorithms are ranked according the segmentation accuracy and execution time. Two metrics are used to calculate the segmentation accuracy; pixel counting and closed contour area overlapping.

The goal of the feature selection is to characterize a lymphocyte class (CLL, or normal) to be recognized by measurements whose values are very similar to lymphocytes in the same cell class, and very different from lymphocytes in different classes. This leads to the idea of seeking distinguishing features that are invariant to transformations of the image such as scale, translation, and rotation. FSAs can be helpful in selection of such features. There are two types of FSAs: filter type and wrapper type. Filter type is simple, however it may not reach dimensional reduction as it cannot rank two features that
capable of totally separate the classes. Wrapper method is – SFS, and SBS – accurate but computationally expensive.

The degree of difficulty of the classification problem depends on the variability in the feature values. Conceptually, the simplest measure of classifier performance is the classification error rate which is percentage of new features that are assigned to the wrong category. The cross-validation is a statistical method of evaluating and comparing learning algorithms by dividing data into two segments: one used to learn or train a model and the other used to validate the model. In typical cross-validation, the training and validation sets must crossover in successive rounds such that each data point has a chance of being validated against.

A complex classifier may allow perfect classification of the training features, however it is unlikely perform well on new test features which is known as over-fitting problem.

In this research five different classifiers are trained and used to classify lymphocyte test images. The classifiers are an SVM, an ANN, a K-NN, a binary decision tree, and an AdaBoost algorithm. The results of the five classifiers are fused together to label a lymphocyte cell as a CLL or normal lymphocyte. The fusion method is the majority voting method with a calculated trust factor using the DST method.

The output of the proposed system is a comprehensive cell analysis report. The report contains a chart for the classifiers along with the fused classification results showing the number of cells classified as CLL or normal lymphocytes. This representation can be used by the hematopathologist to support his/her decision for diagnosis or further analysis.
The no-free-lunch theorem [88] has unarguably proven that there is no such best classifier for all classification problems, and that the best algorithm depends on the structure of the available data and prior knowledge.

The system is installed on a platform containing 8 GB DDR3 RAM, Intel i5-2500K, CPU 3.30GHz Quad Core, 64-bit Windows® 7 operating system, NVidia GeForce 550Ti GPU, and developed using MATLAB® R2011b.

In Chapter 4 the hosting software framework will be presented as well as the SA used to build the system. The proposed tailored wrapper design pattern, which is used to translate the interface of an external software component to the client software interface is presented.
Chapter 4: The Adaptive Framework with a Tailored Wrapper

The segmentation and classification algorithms from the previous chapter were presented in the scope of its mathematical and theoretical background, while its hosting software design was not addressed. In turn, this chapter examines a suitable SA for the proposed system algorithms. This chapter focuses on explaining the methodology that is used to develop an appropriate SA which is adaptable, flexible and reusable.

One of the main requirements of the proposed system architecture is the capability of hosting other algorithms of the kind (i.e. segmentation, FSAs, classification algorithms, etc.) which might have different sets of parameters. For this reason, specific attention has been paid to a new design pattern described in – Section 4.3 – which allows hosting of various algorithms with different sets of parameters.

4.1 Requirements

Adaptive software framework targets a specific problem type and hosts various algorithms to solve it where each of those algorithms are user selectable and may have different set of parameters. For example, the lymphocyte segmentation problem is an algorithmic problem which can be solved by various algorithms, i.e. SVM, Fuzzy C-mean, watershed, graph cut, active contour algorithms, etc. The input to these algorithms is a pointer to an image or a matrix representation of the image, while the output is one or more masks segmented by the algorithm. Another example can be the task of lymphocyte feature classification algorithms. These algorithms assign a label for the features of the test lymphocyte image. Various algorithms have been proposed and they have different
advantages and disadvantages. Some of these algorithms are SVM, ANN, Bayesian Belief Network (BBN), KNN, and decision tree, etc.

This section defines the problem by listing the set of requirements for the software framework and the SA for the proposed CDSS which aims to enhance the system flexibility, scalability and reusability.

The problem is defined as building an adaptable software framework on top of a suitable SA to accommodate for the huge data input and data output by various algorithms, moreover the system must be capable of incorporating an adapter design pattern which is a specific type of structure pattern, and its main focus is to allow the usage of new algorithms by the proposed CDSS – client – . The list of the requirements for this architecture is illustrated in Table 4.1.

Table 4. 1– Functional requirements of the proposed CDSS.

<table>
<thead>
<tr>
<th>Functional Requirements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>The system should be able to host a variety of image data in multiple formats and different processing algorithms</td>
</tr>
<tr>
<td>R2</td>
<td>The system must allow for the comparison of different algorithms in the same category for the accuracy and execution time performance.</td>
</tr>
<tr>
<td>R3</td>
<td>Data exchange. The system must allow for data exchange.</td>
</tr>
<tr>
<td>R4</td>
<td>Input images and output masks visualization. The visualization should be independent of any algorithm implementation and the system must be able to visualize the input and the output of any algorithm hosted in the architecture. Various visualizations might exist. (segmentation masks, features distribution, and classification results)</td>
</tr>
<tr>
<td>R5</td>
<td>The user can select and configure the used algorithms</td>
</tr>
<tr>
<td>R6</td>
<td>The system must allow for interfacing and hooking of a new algorithm. The system must allow the user to perform parameters mapping between the client parameters and the new algorithm parameters, and verify the operation of the new algorithm against built-in test procedures</td>
</tr>
</tbody>
</table>
The above mentioned requirements will be addressed in the solution described in the next sections.

4.2 Solution

This section presents an SA solution to the problem defined by the requirements in the previous section. This solution addresses all the requirements presented above with particular emphasis on the flexibility of the architecture. Modifiability refers to the ability to change certain behaviours. For example, the algorithm should be replaced by another algorithm easily. Extensibility refers to extending the existing behaviours of the system, for example, ability to easily access data from other data stores. The components of the proposed software framework are illustrated in Table 4.2.

Table 4.2– The proposed software framework component to achieve the hosting system requirements.

<table>
<thead>
<tr>
<th>Framework component</th>
<th>Requirement matching</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controller Module</td>
<td>Yes. It keeps track of the existing algorithms and invokes the configured algorithms</td>
<td></td>
</tr>
<tr>
<td>Manipulation algorithms</td>
<td>NO; if the user decides to keep the default processing algorithms (segmentation, feature selection, and classification)</td>
<td></td>
</tr>
<tr>
<td>User interface</td>
<td>Yes. It is mandatory to facilitate the user interaction with the system.</td>
<td></td>
</tr>
<tr>
<td>System setup</td>
<td>R5</td>
<td>No if the user decides to keep the default settings for the algorithms. It is responsible for choosing the algorithms for data manipulation.</td>
</tr>
<tr>
<td>Data vault and Exchange</td>
<td>R1,R3</td>
<td>Yes. A huge amount of data results from the manipulating algorithms.</td>
</tr>
<tr>
<td>Data visualization and tabulation</td>
<td>R2, R3,R4</td>
<td>Yes. To visualize the lymphocyte images, the segmented masks; and the classification results</td>
</tr>
<tr>
<td>Add new Algorithm</td>
<td>R6</td>
<td>No if the user decides to keep the default processing algorithms</td>
</tr>
</tbody>
</table>
The chosen SA is the data-centric architecture style, which supports the hosting of a huge amount of data in various forms and hosting of the manipulating algorithms as well. Additionally, the data-centric SA is a perfect choice for extendibility, reusability and modifiability of the system where many algorithms can access the data and return their respective output.

4.2.1 The Proposed system SA

The high level-architecture with all the components is illustrated in Figure 4.1. This architecture follows the module view controller (MVC) architecture, which allows for implementation flexibility, extendibility, modifiability and reusability as follows:

1. User Interface. The system user deals with the system through a user friendly interface and can visualize the processed images and the system results through the visualization and data tabulation.

![Figure 4.1 – The proposed CDSS software architecture.](image-url)
2. The system setup module is there to switch between the manipulations algorithms when the user wants to choose one of the algorithms to be the main algorithm for the data processing.

3. Data vault and exchange is an auxiliary data repository where data can be copied to another source through the controller module and the user interface.

4. Manipulation algorithms, where a new algorithm is being wrapped to suite the client interface, and the new algorithm is being verified against built-in procedures.

5. Add new algorithm

This module composes the tailored design pattern of the system, which enables hooking of a new algorithm into the system and verifies its operations. The tailored design pattern is one of the contributions of this research and will be discussed in more details in the next section (Section 4.3).

6. Controller module

This is the basic module in the system and is mandatory to control every single part of the CDSS. Figure 4.2 shows the components of this module, which encapsulating a data-centric SA that hosts the data manipulated and annotated by every algorithm. This annotation is important to keep track of which data is generated by which algorithm, which can be easily achieved by renaming the resulting image file by a suffix revealing the algorithm nature; for example the cell mask generated by the SVM algorithm will be named – imagename.svmCell.jpg –. The controller module contains a controller component which manages the communication and settings of the enclosed algorithms. One of the most important parts of the module controller is the CLL modeling and parameter estimation, in which if there is enough images representing the CLL cell from
very early stage to very late stage in the disease cycle, the system can categorize a found CLL cell as belonging to which stage. However there are not enough CLL images describing the CLL cells in every stage and, thus, this is one of the system objectives in the long run to collect enough images in every stage and develop a model that can describe it. This component can serve as an assessment tool for the treatment of the CLL disease.

Figure 4. 2– The components of the controller module showing the data-centric architecture and the relationship between the internal components.
4.3 Tailored wrapper design pattern

One of the main objectives of this research is to expand the capabilities of the proposed system to be able to use other algorithms e.g. segmentation, feature extraction/selection and classification algorithms that might enhance the overall performance of the proposed system and this new algorithms may be written in any programming style and format (M, EXE, DLL, and MEX), and this will achieve the reusability, extendibility and scalability of the system as a research tool for the hematopathologist.

MATLAB® is used extensively to develop code for research purposes in many programming style, which can be a plain script, structured functions, or object oriented programming style, which makes it difficult to integrate these types of programming styles without a huge modification.

One possible solution is developing a generic interface which under user control can map the input and output variables between the client software (CDSS) and a new algorithm. This could be achieved using one type of the structural design pattern known as strategy pattern [76]. However it has some limitations; it is only one type of interface – with the same parameters or without any parameters – , usually object oriented programming (OOP), which may not be useful for the proposed system operation, which requires the usage of multiple algorithms to process a test lymphocyte image, and the algorithms, may not be written in OOP.

Study [89] provided a solution to such a problem by defining an extended strategy pattern which is responsible for pulling the different parameters from the new algorithm to the client software, verify the parameters constraints and let the user performs the
parameters mapping. However, this solution is restricted only to OOP programming style which is not always the case when developing code for research purposes in MATLAB®.

The adapter pattern is a design pattern that translates one interface for a class into a compatible interface, which allows classes to work together that normally could not because of incompatible interfaces, by providing its interface to clients while using the original interface. The adapter translates calls to its interface into calls to the original interface, and the amount of code necessary to do this is typically small. The adapter is also responsible for transforming data into appropriate forms.

In this research a tailored wrapper design pattern is proposed which can perceive the interface for e.g. segmentation, feature selection, and classification algorithm and further detects and pulls the input and output parameters. The user can then map the parameters to the interface of the client (CDSS).

This will facilitate the reusing of the newly added algorithm with minimum user interaction by providing a permanent warp for the new algorithm, which facilitates the cloning of the new added algorithm into the system, in which the user does not have to invoke the tailored pattern every time the same algorithm is used by the client. One of the major concern of this proposed wrapper is to make sure that the hooked algorithm has no internal constrains and defects which may affect the whole operation, which can be achieved by validating the operation of the new algorithm against built-in test procedures for different algorithm before finally hooking the algorithm into the system. For example a new segmentation algorithm can be verified against a known test image and the resulting masks can be examined for the dimension, the data type, and the segmentation accuracy obtained.
In the following section, the proposed design pattern specification is described using a consistent format [76].

4.3.1 Tailored design pattern specification

- Pattern Name and Classification
  The name of the pattern used in this study is “tailored wrapper” and it is classified as a structural design pattern.

- Intent
  The tailored wrapper is used to translate the interface between the client software (CDSS) and the new added algorithm e.g. segmentation, feature selection/extraction, and classification algorithm. The tailored wrapper pulls the concrete algorithm parameters and makes it available to the user to match it with the client interface, and it verifies the interface against a built-in test procedures. The design issue that the proposed pattern addresses is the newly added algorithm programming style. The newly added algorithm may not be only written in OOP, and it may take one of the following: MATLAB® plain script/function, DLL, MEX, and EXE.

- Also Known As
  The tailored wrapper design pattern is known as wrapper.

- Motivation
  There are software components that are available for reuse; however modifications to the components are required to integrate the new concrete software component into a system because of the different interface used. This
could introduce time latency in software development especially if a large number of software components are needed to be integrated into the system.

- **Applicability**

The proposed pattern can be applied whenever the system user requires adding a new algorithm. The pattern can warp a plain MATLAB® script into the system interface by searching for all the possible variables and let the user performing the variables matching to match the client (CDSS) interface.

- **Structure**

Figure 4.3 shows the proposed tailored wrapper design pattern. An example communication scenario is explained showing the design of the proposed tailored wrapper pattern along with the sequence diagram (SD) of the communication scenario.

![Diagram](image)

Figure 4.3 – The proposed tailored wrapper design pattern.
Communication scenario

The tailored wrapper design pattern is designed to enable the proposed system (Client) to communicate to an external algorithm package, which has a different interface. The user can request adding a new algorithm using the tailored wrapper, in which the wrapper identifies the algorithm type, and pulls the parameters from the interface to the user to carry out the matching between the client algorithms and the new algorithm. The communication between the client and the new algorithm can be defined in the following scenario where client, tailored wrapper, and the concrete algorithm roles are defined. The sequence diagram (S.D) showing this communication scenario is illustrated in Figure 4.4.

![Sequence Diagram](image)

Figure 4.4– A sequence diagram showing the interaction between the client and the new algorithm to map the concrete interface of the new algorithm to the client interface.
**Communication Scenario for adding and validate a new algorithm**

**Role of the Client:**

The role of the Client in the communication scenario when using the tailored wrapper is to keep track of the hooking point of the new algorithm, and to remember the interface required by the concrete algorithms attached to the client at the hooking point, which is useful when the parameters matching and typecasting are required. The client invokes the tailored wrapper to insert a new algorithm into the system manipulation algorithm component by requiring the wrapper to get the interface parameters from the new algorithm, then the client displays the interface parameters to the user and the user can match these parameters with the concrete algorithm interface parameters. The parameters are then sent to the wrapper with test values to test the new algorithm. At this point the client can ask the wrapper to warp the new algorithm interface with the mapped parameters, to create a clone of this new algorithm in the manipulation algorithm component of the system, which facilitates the reusing of this new algorithm without repeating the parameters mapping when re-invoking of the same algorithm. The client can further validate the operation of the new algorithm by requiring the tailored wrapper to execute a validation subroutine which is designed for the proposed system algorithms (segmentation, feature selection and classification algorithm). The output of the new algorithm is sent back to the client through the wrapper to visualize and/or tabulate the results for the user.
Role of the concrete algorithm

The new algorithm contains a concrete routine. The routine may have various parameters in its declaration and they can vary depending on the concrete algorithm type. These parameters in the declaration should be mirrored by the list of appropriate parameters.

Role of the tailored wrapper

The role of the proposed design pattern is to manage the interface translation between the new concrete algorithm and the client to add new capabilities to the system. The tailored wrapper can pull the parameters from any algorithm written in MATLAB® in a plain MATLAB® script, MATLAB® function or OOP style. Also the wrapper can handle algorithms written in DLL library, MEX, and executable format.

- Participants
  The CDSS participates as the client software and the concrete algorithm as the adaptee algorithm.

- Collaborations
  See communication scenario.

- Consequences
  This client may require write access to some variables of the new added algorithm (class attributes), which may not be possible because new algorithm may restrict this operation. To solve this problem the wrapper may copy the write protected variables to new variables and uses it for read and write operations.
• Implementation

The proposed wrapper is implemented in MATLAB®, and it can interface to the algorithms that are implemented in MEX, DLL, and EXE. Other language such as JAVA is not supported.

• Sample code and example of implementation

In this example, there are two algorithms which solve the same problem. The nature of the algorithm problem is the lymphocyte cell segmentation. A description and flowchart of the algorithm used to segment the lymphocyte cells are presented in studies [99][100]. The segmentation algorithm proposed in [99] is implemented in plain MATLAB® script without – function body structure– and the segmentation algorithm proposed in [100] is implemented in MATLAB® as a class.

Example code of segmentation class used by the tailored wrapper to add and execute any cell segmentation algorithm:

```matlab
classdef Segmentation

    properties
        Name = '';
    end

    methods
        function obj = Segmentation(inName)
            obj.Name = inName;
        end
        % More methods go in here specific to data
    end
end
```
Tailored wrapper design pattern example code to add and execute a new segmentation algorithm:

classdef TailoredWrapperType
    properties
    end
    methods (Abstract)
        function ValidateExecution (Algorithm, Image) end
        function Getparam() end
        function Warp() end
        function Loadlib() end
        function Loadfunc ()end
    end
    methods (Static)
        function parameters=Getparam(Algorithm)
            parameters[].Name=Names ;
        end

        function [Etime,TotalRes,imCyto,imCell,imNuc,maskCell,maskNuc,maskCyto] = newType(Algorithm, Image)
            switch lower(Algorithm)
                case 'watershed'
                    rslt = watershed;
                case 'svmSeg'
                    rslt = svmSeg;
                otherwise
                    % do some error checking
            end
        end
    end
end
Example code to invoke the new added algorithm at the client side:

Segmentation(Image);

s = TailoredWrapperType (‘watershed’);

[ETIME, TotalRes, imCyto, imCell, imNuc, maskCell, maskNuc, maskCyto] =
ValidateExecution(Algorithm, Image);

s = TailoredWrapperType (‘svmSeg’);

[ETIME, TotalRes, imCyto, imCell, imNuc, maskCell, maskNuc, maskCyto] =
ValidateExecution(Algorithm, Image);

- **Known Uses**

  In real systems the proposed pattern could resemble a transformer, which adapts household electric current from high voltage (100 to 240 volts AC) to low voltage suitable for consumer electronics.

- **Related Patterns**

  The strategy design pattern is closely related to proposed pattern. The important difference between the strategy pattern and the tailored wrapper pattern is that the strategy pattern is built to translate only one type of concrete algorithm with the same parameters or without any parameters, whereas the tailored wrapper is designed to translate different types of algorithm interfaces with different parameters. The proposed pattern can work with facade pattern, which provides a simplified interface to a larger body of code, such as a class library. Moreover the proposed pattern can be used with the decorator design pattern which is used to dynamically add responsibility to the interface by wrapping the original code.
The proposed wrapper can optimize the parameters by doing typecasting or inform the user about the parameters and the user can accept or decline the hooking of this new algorithm. This situation resembles the certificate test, in which a component of software can be accepted as a whole or rejected. This can address the reliability of the system by rejecting the unsuitable component that may increase the downtime of the system (non-functional requirement).

4.4 Conclusion

In this chapter the adaptive software framework that hosts the proposed system is described, which requires a method to add a new algorithm, which addresses the extendibility, modifiability, reusability, and flexibility of the proposed CDSS system. Adding a new concrete algorithm to an existing system suffers from the interface incompatibility, resulting from writing the algorithms in a concrete interface. This problem can be solved by utilizing the proposed tailored wrapper design pattern, which can translate the new algorithm interface to the client compatible interface, which can be achieved by pulling the new concrete algorithm parameters, and presenting it to the user to match these parameters with the expected parameters by the client.

The wrapper contains different methods to pull the parameters from the new algorithms, and can verify the new algorithms operation by receiving test variables from the client, sending it to the new algorithms; and executing the new algorithms. The results are then visualized by the client and presented to the user. The wrapper can permanently warp the new algorithms to clone them into the system.
Chapter 5: Results and Discussion

In this chapter I present and discuss the results of the proposed system. First the segmentation results are presented and discussed for the three different segmentation algorithms along with every experiment setup. Features extraction and selection is presented and discussed. The classification algorithm results are presented in the following manner; first the results of the training process of the classifier models followed by the model parameters tuning and the final classifier model results. More classification results are discussed in terms of classification accuracy, receiver operating characteristics (ROC), sensitivity and specificity along with the fusion results of the classifier models with the a trust factor calculated using the DST. The results of the reporting tool are also presented. This chapter is concluded with a comparison between the proposed system and other available CDSS used to detect CLL.

5.1 Lymphocyte cell segmentation using Watershed algorithm and Otsu’s method

In this experiment, 140 CLL and normal lymphocyte images are used to assess the performance of the watershed based segmentation algorithm. The algorithm could process only 132 images as the remaining (8 images) were for lymphocytes that were tightly tied to RBCs. Figure 5.1 shows the resulting output masks, the manually segmented masks of the cell, the nucleus, and the cytoplasm, which are used to estimate the accuracy of the segmentation algorithm. Figure 5.1.b shows the segmented cell mask which is compared to the manually segmented cell mask shown in Figure 5.1.c for cell segmentation accuracy. The segmented nucleus mask shown in Figure 5.1.d is compared
to the manually segmented mask shown in Figure 5.1.e for nucleus segmentation accuracy.

Figure 5. 1– Sample images (CLL and normal) and the resulting output segmentation masks (a) Original images, (b) Extracted cell mask, (c) Manually segmented cell, (d) Extracted nucleus mask, (e) Manually segmented nucleus, (f) Extracted cytoplasm mask.

Figure 5.1.f shows the cytoplasm mask resulting from pixel-to-pixel subtraction of the nucleus mask from the cell mask. Figure 5.2 shows the effect of removing 1% of the local minima before applying the watershed algorithm which shows a reduction in
over and under segmentation errors, when suppressing 1% of the local minima. Increasing the local minima suppression, results in merging of the lymphocyte with the nearby RBCs. The result demonstrates the following segmentation accuracy: 98.41 % for the nucleus, 99.99% for the whole cell and 99.96% for the cytoplasm. Table 5.1 shows the segmentation results using the pixel counting and closed contour area overlapping methods.

<table>
<thead>
<tr>
<th>Original image</th>
<th>0% local minima suppression (under-segmentation)</th>
<th>1% local minima suppression (over-segmentation)</th>
<th>10% local minima suppression (over-segmentation)</th>
<th>30% local minima suppression (over-segmentation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) CLL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>% CLL Cell segmentation Accuracy</td>
<td>50.12 % under-segmentation error</td>
<td>95.24 % Cell segmentation accuracy</td>
<td>23.75 % over-segmentation error</td>
<td>39.05 % over-segmentation error</td>
</tr>
<tr>
<td>(b) Normal lymphocyte</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>% Normal Cell segmentation Accuracy</td>
<td>61.39 % under-segmentation error</td>
<td>87.31 % Cell segmentation accuracy</td>
<td>15.99 % over-segmentation error</td>
<td>46.94 % over-segmentation error</td>
</tr>
</tbody>
</table>

Figure 5.2– Sample images (CLL and normal) and the resulting output with increasing the percentage suppression of the local minima (a) Sample CLL image (b) Sample normal lymphocyte image.
Table 5.1– Segmentation accuracy of the watershed algorithm using two different metrics: pixel counting and closed contour area overlapping (Cyto=Cytoplasm).

<table>
<thead>
<tr>
<th>Watershed segmentation based method</th>
<th>Pixels counting</th>
<th>Closed contour Area overlapping</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cell</td>
<td>Nucleus</td>
</tr>
<tr>
<td>Min</td>
<td>88.31</td>
<td>88.31</td>
</tr>
<tr>
<td>Avg</td>
<td>97.08</td>
<td>96.90</td>
</tr>
<tr>
<td>Max</td>
<td>99.99</td>
<td>98.41</td>
</tr>
<tr>
<td>Std</td>
<td>±1.31</td>
<td>±0.95</td>
</tr>
</tbody>
</table>

5.2 Lymphocyte cell segmentation using Machine learning algorithms and K-means algorithm

The SVM and the ANN classifiers can process all the 140 pre-classified images used in this experiment, even if the lymphocyte cells were tightly tied to RBCs, thereby the problems of over and under segmentation are reduced significantly. The pixels’ color of the test image is normalized to reduce the effect of the cell and the nucleus color variations. The SVM and the ANN classify the pixels as belonging to the nucleus, the cell or the background. Figure 5.3 shows the resulting output masks and the manually segmented masks for the cell and the nucleus. Figure 5.3.b shows the segmented cell mask which is compared to the manually segmented cell mask shown in Figure 5.3.c and Figure 5.3.d for cell segmentation accuracy. Also the segmented nucleus mask shown in Figure 5.3.e is compared to the manually segmented mask shown in Figure 5.3.g for nucleus segmentation accuracy. Figure 5.3.h and 5.3.i show the cytoplasm mask resulting from pixel-to-pixel subtraction of the nucleus mask from the cell mask. Figure 5.3.j shows the resulting cell mask for un-normalized pixels’ color values using the same SVM classifier.
<table>
<thead>
<tr>
<th>(a) Color Image</th>
<th>(b) Segmented Cell Mask (SVM)</th>
<th>(c) Segmented Cell Mask (ANN)</th>
<th>(d) Manually Segmented Cell</th>
<th>(e) Segmented Nucleus Mask (SVM)</th>
<th>(f) Segmented Nucleus Mask (ANN)</th>
<th>(g) Manually Segmented Nucleus</th>
<th>(h) Cytoplasm Mask (SVM)</th>
<th>(i) Cytoplasm Mask ANN</th>
<th>(j) Cell Output Mask without Pixel color Normalization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C. L. L. 1</td>
<td>Normal 1</td>
<td>C. L. L. 2</td>
<td>Normal 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C. L. L. 1</td>
</tr>
</tbody>
</table>

Figure 5. 3– Sample images (C. L. L. and normal) and the resulting output masks (a) Original images, (b) Extracted cell mask using an SVM, (c) Extracted cell mask using an ANN (d) Manually segmented cell, (e) Extracted nucleus mask using an SVM, (f) Extracted nucleus mask using an ANN (g) Manually segmented nucleus, (h) Extracted cytoplasm mask using an SVM, (i) Extracted cytoplasm mask using an ANN, (j) Cell output mask without pixel color normalization (the SVM method).
Table 5.2 and Table 5.3 show the resulting accuracy measurement using the two accuracy measurement metrics for the ANN based segmentation method and the SVM based segmentation method. Using pixels counting metric, the proposed SVM segmentation algorithm is capable of extracting the lymphocyte nucleus with average accuracy 97.0% and 92.08% accuracy for the cytoplasm. Using the overlapping contour metric, the average segmentation accuracy for the nucleus is 94.38% and for the cytoplasm is 60.92%. Table 5.4 shows a comparison between studies [24][28] and the proposed segmentation methods based on the SVM, the ANN, and the watershed algorithm.

Table 5.2– The Percentage segmentation accuracy measurement for 140 lymphocyte images segmented by the ANN algorithm which is evaluated by two different methods: Pixel counting and Area overlapping (Cyto=Cytoplasm).

<table>
<thead>
<tr>
<th>The ANN based segmentation method</th>
<th>Pixels counting</th>
<th>Area overlapping</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cell</td>
<td>Nucleus</td>
</tr>
<tr>
<td>Min</td>
<td>89.71</td>
<td>94.22</td>
</tr>
<tr>
<td>Avg</td>
<td>97.51</td>
<td>96.58</td>
</tr>
<tr>
<td>Max</td>
<td>98.61</td>
<td>99.99</td>
</tr>
<tr>
<td>Std</td>
<td>±0.79</td>
<td>±0.72</td>
</tr>
</tbody>
</table>

Table 5.3– The Percentage segmentation accuracy measurement for 140 lymphocyte images segmented by the SVM algorithm which is evaluated by two different methods: Pixel counting and Area overlapping (Cyto=Cytoplasm).

<table>
<thead>
<tr>
<th>The SVM based segmentation method</th>
<th>Pixels counting</th>
<th>Area overlapping</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cell</td>
<td>Nucleus</td>
</tr>
<tr>
<td>Min</td>
<td>89.95</td>
<td>95.67</td>
</tr>
<tr>
<td>Avg</td>
<td>97.62</td>
<td>97.00</td>
</tr>
<tr>
<td>Max</td>
<td>98.69</td>
<td>98.43</td>
</tr>
<tr>
<td>Std</td>
<td>±0.77</td>
<td>±0.50</td>
</tr>
</tbody>
</table>
Table 5.4—Comparison of the proposed segmentation method evaluated by the closed contour overlapping area and other studies (Nuc=nucleus, Cyto=cytoplasm, over-seg=over-segmentation, under-seg=under-segmentation).

<table>
<thead>
<tr>
<th>Study</th>
<th>% Nuc</th>
<th>% Cyto</th>
<th>% Cell Over-Seg</th>
<th>% Cell Under-Seg</th>
<th>Occlusion</th>
<th>Color variations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guo, et al. [24]</td>
<td>Up to 94</td>
<td>Up to 95</td>
<td>Not Argued</td>
<td>Not Argued</td>
<td>Not Argued</td>
<td>Suffer</td>
</tr>
<tr>
<td>Madhloom, et al. [28]</td>
<td>Up to 95</td>
<td>N.A</td>
<td>Not Argued</td>
<td>Not Argued</td>
<td>Not Argued</td>
<td>Suffer</td>
</tr>
<tr>
<td>Proposed SVM based Method</td>
<td>Up to 98.49</td>
<td>Up to 86.57</td>
<td>0-63.37</td>
<td>2.07-18.43</td>
<td>Eliminated</td>
<td>Robust</td>
</tr>
<tr>
<td>Proposed ANN based Method</td>
<td>Up to 97.46</td>
<td>Up to 82.66</td>
<td>0-63.34</td>
<td>0.924-19.31</td>
<td>Eliminated</td>
<td>Robust</td>
</tr>
<tr>
<td>Proposed watershed algorithm</td>
<td>Up to 98.69</td>
<td>Up to 89.0</td>
<td>0-100</td>
<td>0.145-100</td>
<td>Exist problem</td>
<td>Suffer</td>
</tr>
</tbody>
</table>

From Tables 5.1, Table 5.2, and Table 5.3; it can be concluded that the pixel counting metric shows a more reliable representation for the cytoplasm accuracy measurement, due to some lymphocyte images have scattered pixels in their cytoplasm masks, which fail to compose a closed contour, and, thus, resulting in slight correlation with the manually segmented cytoplasm mask using the closed contour area overlapping. The closed contour area overlapping metric is a better representation for the cell and nucleus segmentation accuracy because the cell and nucleus masks can compose closed contours, which are used to estimate the segmentation accuracy.
5.3 Performance comparison

Figure 5.4 and Figure 5.5 show the segmentation accuracy comparison chart for the three algorithms used by the proposed system to segment a lymphocyte test image. Figure 5.4 illustrates the nucleus under segmentation error resulting from using the proposed segmentation methods, in which the SVM exhibits the smallest under segmentation error, and, this is because the separation hyper-plane generated from the SVM training phase tends to maximize the margin between the cell pixels and the background pixels. The ANN based segmentation algorithm comes second as the network topology use only 10 neurons in its single hidden layer to speed up the segmentation process, however this leads to increase the over-segmentation and under-segmentation error slightly more than that of the SVM. The watershed algorithm has the largest under-segmentation and over-segmentation error due to the thick watershed lines resulting from the watershed transform of the gray scale image.

Figure 5.5 shows the cell segmentation accuracy resulting from the three segmentation algorithms, in which the watershed algorithm poorly performs when the lymphocyte cells are tight to the surrounding RBCs – occlusion problem – the ANN shows a better performance than the watershed algorithm, while the SVM shows a superior performance in which the occlusion problem is no longer exist.
Figure 5.4– Nucleus under-segmentation error of the three proposed segmentation methods: the SVM, the ANN, and the watershed algorithm.

Figure 5.5– Cell accuracy estimation for 140 test images using the three proposed segmentation methods based on an SVM, an ANN, and a watershed algorithm.

The execution time of the segmentation algorithms are recorded; which represent the time required by any algorithm to extract the cell from the complicated background and further divide the cell into nucleus and cytoplasm masks. The execution time is measured using Intel® quad core CPU i5 2.53GHz, 4GB DDR3 RAM PC Windows® 7
64-bit using MATLAB® 2011b. Figure 5.6 shows the execution time for every algorithm for processing the 140 manually segmented test lymphocyte images, in which the SVM algorithm demonstrates the lowest processing time, as the SVM segmentation requires only finding the side of the hyper-plane at which every pixel should reside [100]. The ANN requires the substitution of the pixel color values into the network model to find its category, which may increase the processing time. The watershed has the largest processing time as it is a global transform and the algorithm requires pre-processing and post-processing using morphological operations [99]. Every algorithm consumes a little bit more time for the first image as it requires loading the classifier model. According to the aforementioned performance, the system selects the SVM algorithm to be the main segmentation algorithm.

![Figure 5.6](chart.png)

Figure 5.6– The execution time of the three proposed segmentation algorithms for the 140 lymphocyte test images.

The Otsu’s method can be used to segment the nucleus; however the best results are dependent on the assumptions made by the Otsu’s algorithm which are bimodal
histogram of the images and uniform illumination conditions. The results show that the watershed algorithm can be used to segment the cell; however it suffers from over and under segmentation error due to the variation of the local minima in the watershed distance transform. That is why 1% suppression of the local minima is required for better performance, however the amount of local minima suppression is totally subjective to the image quality and the degree of occlusion of the lymphocyte cells to the surrounding RBCs, for instance when the lymphocyte cell is tight to the surrounding RBCs, the watershed algorithm fails to segment the cell as the cell is washed away in the background.

Comparing with other researches, the proposed segmentation method based on watershed algorithm has introduced some enhancements in the segmentation of lymphocytes. The maximum segmentation accuracy achieved by [26] was 92% for the nucleus and 78% for the cytoplasm, whereas the proposed method gives 98.43% maximum accuracy for the nucleus and 99.85% maximum accuracy for the cytoplasm.

The SVM and the ANN classification algorithms can be used to identify the pixels that belong to a lymphocyte cell from a complicated background in a microscopic image. The number of cytoplasm pixels colour trapped inside the nucleus can introduce high degree of cell and nucleus colour variations. The results show that the proposed methods based on the machine learning algorithms are robust to the variations of the cell and nucleus colour conditions; as the training and test images colour pixels are normalized.

The normalization step is of much importance as recommended by [90]. The normalization stage helps decrease the colour disparities of the cell and nucleus which makes the algorithm robust to the cell and the nucleus colour variations. Using the features
(pixel colour) normalization and k-means clustering algorithm provide the proposed method with discriminative power, which yields high segmentation accuracy.

Closed contour area overlapping metric shows a low performance of the cytoplasm segmentation, as most of the cells have scatter pixels of cytoplasm which have no closed contour to form a cytoplasm area. However the pixel counting metric is capable of estimating the cytoplasm segmentation accuracy. Pixel counting segmentation accuracy metric has some limitation to precisely estimate the accuracy of the masks with high scattered pixels distributed in the background of the mask.

Study [24] used an SVM for the nucleus segmentation and showed accuracy up to 94% while the proposed method yields up to 98.43%. Study [28] shows up to 95% for cytoplasm segmentation accuracy while the proposed method yields up to 99.85%. The results show that the application of the SVM and the ANN classifiers with k-means algorithm utilizing the normalization of the input pixel colours are of high accuracy in lymphocyte cell segmentation. Study [38] manages to achieve an accuracy of 90-95% in restoring the lymphoblast pixels from the original image. The proposed methods based on the machine learning algorithms yields an enhancement in cell segmentation.

Considering the results of the MLA based methods; the occlusion problem is totally eliminated. Figure 5.4 shows the nucleus segmentation accuracy using closed contour area overlapping metric with the under-segmentation error for every image. Image number 37 (red dot circle) has a high under-segmentation error as it has been misclassified by the ANN algorithm.

The SVM, the ANN, and the watershed based segmentation algorithms are highly tuned to segment only the nucleus and cytoplasm of the lymphocytes (CLL and normal).
5.4 Feature selection

The feature selection aims to reduce the dimensionality of the features used by the system classifiers in order to speed up the classification process. By visual inspection of the mean and standard deviation of the extracted features; it can be concluded that some features have larger mean and standard deviation than the others, which are (F8, F9, F10, F18, F19, F20, F21, F28, F29, F30 and F31) (See Table 3.2). Dimensional reduction can be achieved by removing these features; and, thus, the remaining features will have equal influence in the classification process.

The second step in the features selection process is to test the separation power of these features, which is achieved by using an acquired software component [83], which is based on the wrapper method, which uses the Naïve Bayes network classifier and – SFS and SBS – to select the best features. Figure 5.7 shows the total extracted 31 features means and standard deviation, for the CLL and normal lymphocytes, while Figure 5.8 shows the selected features after removing the high mean and standard deviations features. The physical interpretation of these features is illustrated in Table 3.2. Figure 5.9 shows the features selection based on the SBS wrapper method which suggest removing only feature F1 which is cell eccentricity measurement; however Figure 5.10 suggests keeping it based on the SFS wrapper method. By examining the CCR curve of both results it can be concluded that keeping the F1 increases the lower limit of the CCR. The selected features are used to train the proposed system classifiers.
Figure 5. 7– Distribution of the mean and standard deviation of the CLL and normal lymphocyte features.

Figure 5. 8– Distribution of the mean and standard deviation of the selected CLL and normal lymphocytes features after removing the high mean and standard deviation features.
Figure 5. 9– SBS wrapper external software component used to select the classifiers optimum features.

Figure 5. 10– SFS wrapper external software component used to select the classifiers optimum features.
5.5 Lymphocyte cell Classification

The proposed system uses five different classifiers to classify a lymphocyte cell as CLL or normal. In this section of the results I present the results of the classifiers in the training phase, followed by the selection of the best model parameters for the five classifiers and the tuning process with the reinforcement learning procedures. The Receiver Operating characteristics (ROC) analysis for the CLL cell type is also presented. The parameters determining the performance of the classifiers are outlined and discussed.

5.5.1 Classifier models training performance

Table 5.5 shows the resulting confusion matrix for every classifier model. The confusion matrix, which resulted from the training process, shows how the predictions are made by every model on the training dataset. The rows correspond to the known class of the data (manual classification). The columns correspond to the predictions made by every model. The value of each element in Table 5.5 is the percentage of predictions made with the class corresponding to the row. Thus, the diagonal elements show the percentage of the correct classifications made for each class, and the off diagonal elements show the errors made by every classifier model. Table 5.6 shows the confusion matrix resulting from the fusion of the SVM, KNN, and Decision Tree, which shows a better performance than the fusion of all classifier models.

<table>
<thead>
<tr>
<th>SVM</th>
<th>ANN</th>
<th>KNN</th>
<th>Decision Tree</th>
<th>AdaBoost</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLL</td>
<td>Nor</td>
<td>CLL</td>
<td>Nor</td>
<td>CLL</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLL</td>
<td>96.12</td>
<td>3.88</td>
<td>89.92</td>
<td>100</td>
</tr>
<tr>
<td>Nor</td>
<td>6.1</td>
<td>93.9</td>
<td>20.73</td>
<td>79.27</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLL</td>
<td>93.0</td>
<td>7.0</td>
<td>96.12</td>
<td>3.88</td>
</tr>
<tr>
<td>Nor</td>
<td>6.1</td>
<td>93.9</td>
<td>23.17</td>
<td>76.83</td>
</tr>
</tbody>
</table>
Table 5.6—Confusion matrix of the three models (SVM, KNN, and D.Tree) fusion and all models fusion representing the performance of every composite fused model classifying the training dataset (The numbers are presented as percentage).

<table>
<thead>
<tr>
<th>Fusion of (SVM, ANN, KNN, Decision Tree, and AdaBoost)</th>
<th>Fusion of (SVM, KNN, and Decision Tree)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLL</td>
<td>Nor</td>
</tr>
<tr>
<td>CLL</td>
<td>98.45</td>
</tr>
<tr>
<td>Nor</td>
<td>4.88</td>
</tr>
</tbody>
</table>

Table 5.7 shows the classification performance attributes results for every single classifier model along with the fusion results of the SVM, KNN, and decision tree using the majority-voting method. Figure 5.11 illustrate the ROC curves of the trained classifier models and the fused results.

Table 5.7—Comparison between the classifiers performance attributes on the training dataset with the fusion results using the majority-voting fusion method. The segmentation method used is the SVM based segmentation method.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>%Sensitivity</th>
<th>%Specificity</th>
<th>%FPR</th>
<th>%Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>96.12</td>
<td>93.9</td>
<td>6.1</td>
<td>95.0</td>
</tr>
<tr>
<td>ANN</td>
<td>89.92</td>
<td>79.27</td>
<td>20.73</td>
<td>84.6</td>
</tr>
<tr>
<td>KNN</td>
<td>100</td>
<td>100</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>D.Tree</td>
<td>93.0</td>
<td>93.9</td>
<td>6.1</td>
<td>93.46</td>
</tr>
<tr>
<td>AdaBoost</td>
<td>96.12</td>
<td>76.83</td>
<td>23.17</td>
<td>86.48</td>
</tr>
<tr>
<td>Fusion of all classifier</td>
<td>98.45</td>
<td>95.12</td>
<td>4.9</td>
<td>96.79</td>
</tr>
<tr>
<td>Fusion of SVM, KNN, D.Tree</td>
<td>99.22</td>
<td>98.78</td>
<td>1.22</td>
<td>99.0</td>
</tr>
</tbody>
</table>


Figure 5.11—Receiver Operating Characteristics of the trained classifier models used by the proposed system with the SVM based segmentation algorithm.

Figure 5.12 and Figure 5.13 show the reports generated by the proposed system classifying the training dataset.
Figure 5.12– Output report generated by the proposed system for the CLL training images.

Figure 5.13– Output report generated by the proposed system for the Normal training images.
5.5.2 The SVM classifier

The training algorithm of the SVM classifier model searches for the radial basis function (RBF) sigma and the box constraint values which maximize the CCR and minimize the misclassified samples. The algorithm used to extract these parameters is shown in Figure 3.6. The results of this algorithm are illustrated in Table 5.8 which reflects the repeated search process for the local minima in the space for the possible values of sigma and the box constraint. The search is repeated 10 times one for each fold of the cross-validation process. The search process uses the optimization technique of multidimensional unconstrained nonlinear minimization (Nelder-Mead) [91].

Table 5.8 – The local minima of the search space values for the RBF sigma and the box constraint.

<table>
<thead>
<tr>
<th>Local Minima</th>
<th>0.34</th>
<th>0.19</th>
<th>0.26</th>
<th>0.59</th>
<th>0.21</th>
<th>0.23</th>
<th>0.36</th>
<th>0.37</th>
<th>0.26</th>
<th>0.27</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sigma</td>
<td>0.58</td>
<td>6.8</td>
<td>0.94</td>
<td>0.35</td>
<td>4.73</td>
<td>2.17</td>
<td>0.56</td>
<td>0.48</td>
<td>1.33</td>
<td>0.81</td>
</tr>
<tr>
<td>Box constraint</td>
<td>0.73</td>
<td>2.4</td>
<td>3.7</td>
<td>0.70</td>
<td>3.33</td>
<td>1.61</td>
<td>2.16</td>
<td>1.84</td>
<td>3.12</td>
<td>2.250</td>
</tr>
</tbody>
</table>

Table 5.8 shows that there are more than one local minimum that are very close to each other and represent possible candidates for the optimum parameters. To get the best results the concept of reinforcement learning is utilized in which the candidate local minima at (0.19, 0.21, 0.23, 0.26, and 0.27) are used to train an SVM model with the corresponding sigma and box constraint values, and some values outside the boundary of the chosen parameters are used in the reinforcement learning process to check of the goodness of the local minima parameters. The chosen sigma and box constraints values are 1.4796, 5.1185 respectively.

A dataset (testing dataset) of 799 pre-classified lymphocyte images which are 662 images of CLL cells and 137 images of normal cells which are used in the reinforcement
learning process (See Table 3.1), at which the bias parameter of the model is determined, which represents the closeness of the decision plane to a specific cell type (CLL or normal). If the decision plane is close to one specific cell type it would be biased to this cell type. A good decision plane would be equally apart from the two types of cells. The reinforcement learning process is repeated for 100 times and the best run setting is chosen, which is the iteration number 60, which yields the highest accuracy for the two cell class (CLL and normal), as illustrated in Figure 5.14.

Figure 5.14– The SVM reinforcement learning repetition to choose the best parameter.

Figure 5.15 shows the ROC curve of every classifier model using the testing dataset. Figure 5.15 shows the ROC of the SVM classifier, which shows that the SVM has a sensitivity of 85.97%, specificity of 84.02%, false positive rate of 15.98%, and total accuracy of 84.99%. The SVM model considers the features of the CLL cells to be the positive samples and the features of the normal lymphocyte cells to be the negative samples. The sensitivity represents the percentage of the true positive (TP) classified as CLL lymphocyte cells by the algorithm and the specificity represents the percentage of the true negative (TN) classified as normal lymphocyte cells by the algorithm. The false positive rate represents the images classified as CLL while they are Normal. The ROC
Analysis can reflect the performance of the classifier models to predict unseen test lymphocyte images. Table 5.9 shows the classifier performance parameters used to evaluate and compare the classifier models used by the proposed system.

![Figure 5.15](image)

Figure 5.15—Receiver Operating Characteristics for the classifier models used by the proposed system with the SVM based segmentation algorithm.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Definition</th>
<th>Calculation</th>
</tr>
</thead>
<tbody>
<tr>
<td>TP</td>
<td>number of true positives samples</td>
<td></td>
</tr>
<tr>
<td>FN</td>
<td>number of false negatives samples</td>
<td></td>
</tr>
<tr>
<td>FP</td>
<td>number of false positives samples</td>
<td></td>
</tr>
<tr>
<td>TN</td>
<td>number of true negatives samples</td>
<td></td>
</tr>
<tr>
<td>accuracy</td>
<td>Correct classification hits for TP and TN</td>
<td>(TP+TN)/(TP+FN+FP+TN)</td>
</tr>
<tr>
<td>TPR</td>
<td>true positive rate, sensitivity, recall</td>
<td>TP/(TP+FN)</td>
</tr>
<tr>
<td>FPR</td>
<td>false positive rate, fallout</td>
<td>FP/(TN+FP)</td>
</tr>
<tr>
<td>TNR</td>
<td>true negative rate, specificity</td>
<td>TN/(TN+FP)</td>
</tr>
</tbody>
</table>
5.5.3 *ANN classifier*

The training algorithm for the ANN searches for the weights and the bias for a given set of neurons in the single hidden layer of the net using the conjugate gradient descent back propagation algorithm. The training algorithm which outlined in Figure 3.7 increases the number of neurons and finds the corresponding weights and bias for the net topology which aims to maximize the CCR and minimize the misclassified instances. For every net topology the training data is divided randomly into 70% for training, 15% for validation and 15% for testing. The process for calculating the cross-validation process is repeated and the parameters – the weights and bias – for the minimum cross-validation error is chosen. Then the net is used to classify a pre-classified 799 lymphocyte test dataset images in the concept of reinforcement learning and the accuracy is recorded.

The experiment is run for 100 iterations, in which the number of neurons is increased by one and the whole process is repeated. At the last step the net topology that yields the minimum classification error is chosen. As shown in Figure 5.16 the number of neurons that give the maximum accuracy for both classes is 43 neurons. This net topology is selected and used by the proposed system to classify a test lymphocyte. Figure 5.15 shows the ROC curve of the ANN classifier, which shows that the ANN has a sensitivity of 84.32 %, specificity of 82.19%, false positive rate of 17.81%, and total accuracy of 83.26%.
5.5.4 KNN classifier

The parameters for the KNN classifier are the number of the neighbours – k – used to label a test lymphocyte image and the distance metric used to rank the neighbours. The used distance metric, in this research, is the Euclidean distance. To determine the number of k, the training data is divided into 10-fold for test and training. Initially k is set to 1 and the re-substitution and the cross-validation errors are averaged for 10 times. The algorithm repeats the process with increasing k from 1 to 100. Figure 5.17 shows the relation between the re-substitution and cross-validation error with increasing k, in which k=1 and k=2 have the minimum cross-validation error. The optimum value for k is – 1 – as choosing k=2 have the effect of increasing the processing time without any gain in the classification accuracy. In a noisy environment the choice of k may be increased to 3 or 5 first neighbours. Figure 5.15 shows the performance of the K-NN classifier which is used to classify the 799 test lymphocyte images – k=1– which shows that the K-NN classifier has a sensitivity of 80.78%, specificity of 83.1%, false positive rate of 16.9%, and total accuracy of 81.94%.
5.5.5 Decision Tree classifier

Figure 5.18 shows the decision tree model fit to all of the training dataset. The training process of the decision tree aims to get the optimum terminal leaf nodes count. This number is used to prune the tree to speed up the classification process. The leaf node count is determined as illustrated in Figure 5.19 in which the re-substitution and cross-validation errors are used to select the tree terminal leaf nodes. The best choice is a 3 leaf nodes, which is chosen because the re-substitution and cross-validation errors are almost the same, and, this reduces the effect of over-fitting and maximizes the CCR for new unseen test lymphocyte images. The pruned tree is shown in Figure 5.20 in which the splitting node feature is determined by the entropy information as explained in Eq. (3.12).

The splitting features of the training dataset are F13 which represents the nucleus eccentricity and F2 which represents the cell solidity. Figure 5.15 shows the performance of the 3-leaf nodes decision tree classifier which is used to classify the 799 pre-classified test lymphocyte images, which shows that the decision tree classifier has a sensitivity of
72.19 %, specificity of 87.67%, false positive rate of 12.33%, and total accuracy of 79.93%.

Figure 5. 18– Full tree model fitted to the lymphocyte training dataset.

Figure 5. 19– The cross-validation and the re-substitution errors used to determine the number of leaf nodes for the decision tree classifier.
5.5.6 AdaBoost classifier

The AdaBoost classifier depends on scaling the training dataset by the classification error; in which the weak classifier focuses more on misclassified instances, which is achieved by putting more weights on the misclassified samples and less on the truly identified ones. The weak classifier used by the AdaBoost algorithm is the decision tree explained in section 3.5.4, therefore the same number of terminal leaves is used by the AdaBoost decision tree algorithm. The other parameter needed by the AdaBoost is the number of weak learners – classifiers – which is determined by finding the number of trees that have the best CCR, as illustrated in Figure 5.21, in which the classification test results and the cross-validation errors are used to select the parameters – number of weak learners – in which the test curve shows a decreasing in the classification error, and the cross-validation error is closely following the classification error while increasing the number of trees. Choosing the number of learners is a critical step to ensure good classification results and not to bias the aggregation of the results. The chosen number of learners is 285 at which the classification error stops fluctuating and the cross-validation error is fairly small.
Figure 5. 21 – The relationship between the test classification error and the cross-validation error with increasing the number of trees used by the AdaBoost algorithm, which is used to determine the optimum number of trees to be used.

Figure 5.15 shows the performance of the 285 decision trees with 3-leaf nodes per tree used by the AdaBoost classification algorithm which is used to classify the pre-classified 799 test lymphocyte images, which shows that the AdaBoost classifier has a sensitivity of 81.16 %, specificity of 75.8%, false positive rate of 24.20%, and total accuracy of 78.48%.

5.5.7 Fusion of the Classifier models results

The main idea of classifier models results fusion is to compose a composite classifier model with a better CCR from a group of relatively simple classifier models, which is based on the fact that every classifier may make different mistakes, in the classification process, however if the classifiers made exactly the same mistakes; using any fusion method will not yield any better performance and may lead to classification biasing toward a specific cell class. In this research the majority voting fusion method is used with the DST calculated trust factor, and training the classifier models using the same
training dataset. Table 5.10 shows the classification performance attributes results of every single classifier model along with the fusion results using the majority-voting method. Figure 5.15 illustrate the ROC curves of the fused results, and Figure 5.22 shows a visual comparison of the classification performance attributes between the different classifier models and the fused version using the majority voting method.

![Figure 5.22](image)

Figure 5. 22– The classification performance attributes for the different classifier models and the fused model results using the majority voting method.

Table 5. 10– Comparison between the classifiers performance attributes with the fused results using the majority-voting fusion method. The segmentation method used is the SVM based segmentation method and the execution time is the average execution time for 1010 images.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>% sensitivity</th>
<th>% specificity</th>
<th>% false positive rate</th>
<th>% total accuracy</th>
<th>Process(Segmentation, features extraction, and classification) Execution Time/image (Sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>85.96</td>
<td>84.02</td>
<td>15.98</td>
<td>84.99</td>
<td>0.246</td>
</tr>
<tr>
<td>ANN</td>
<td>84.32</td>
<td>82.19</td>
<td>17.81</td>
<td>83.26</td>
<td>0.247</td>
</tr>
<tr>
<td>KNN</td>
<td>80.78</td>
<td>83.1</td>
<td>16.89</td>
<td>81.94</td>
<td>0.244</td>
</tr>
<tr>
<td>D.Tree</td>
<td>72.19</td>
<td>87.67</td>
<td>12.33</td>
<td>79.93</td>
<td>0.248</td>
</tr>
<tr>
<td>AdaBoost</td>
<td>81.16</td>
<td>75.8</td>
<td>24.2</td>
<td>78.48</td>
<td>0.328</td>
</tr>
<tr>
<td>Majority Voting fused model</td>
<td>84.95</td>
<td>89.5</td>
<td>10.5</td>
<td>87.23</td>
<td>0.0016*10⁻³</td>
</tr>
</tbody>
</table>
The system offers fusing of any number of classifier models, however choosing the optimum combination of the classifier models can guarantee a faster and better classification performance, which can be achieved by using the DST to calculate a trust factor for every possible classifier ensemble composite from the classifier models existing in the system.

The ROC demonstrated in Figure 5.15 is based on the SVM segmentation algorithm, which is used by the proposed system to segment the lymphocyte cell as described in section 5.2. Other segmentation methods can be used by the proposed system and the associated ROC figures are illustrated in Figure 5.23 for the ANN segmentation based method and Figure 5.24 for the watershed segmentation based method. Moreover Table 5.11 and Table 5.12 show the classification performance attributes results of every single classifier model along with the fusion results using the majority-voting method for the ANN based segmentation method and the watershed based segmentation method respectively.
Table 5.11 – Comparison between the classifiers performance attributes with the fused results using the majority-voting fusion method. The segmentation method used is the ANN based segmentation method and the execution time is the average execution time for 1010 images.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>% sensitivity</th>
<th>% specificity</th>
<th>% false positive rate</th>
<th>% total accuracy</th>
<th>Process (Segmentation, features extraction, and classification) Execution Time/image (Sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>86.60</td>
<td>74.88</td>
<td>25.12</td>
<td>80.74</td>
<td>0.256</td>
</tr>
<tr>
<td>ANN</td>
<td>84.20</td>
<td>79.45</td>
<td>20.55</td>
<td>81.82</td>
<td>0.258</td>
</tr>
<tr>
<td>KNN</td>
<td>80.02</td>
<td>80.36</td>
<td>19.63</td>
<td>80.20</td>
<td>0.264</td>
</tr>
<tr>
<td>D.Tree</td>
<td>77.12</td>
<td>81.74</td>
<td>18.26</td>
<td>79.43</td>
<td>0.261</td>
</tr>
<tr>
<td>AdaBoost</td>
<td>81.79</td>
<td>76.26</td>
<td>23.75</td>
<td>79.03</td>
<td>0.337</td>
</tr>
<tr>
<td>Majority Voting fused model</td>
<td>85.34</td>
<td>81.28</td>
<td>18.72</td>
<td>83.31</td>
<td>0.0016*10^{-3}</td>
</tr>
</tbody>
</table>

Figure 5.23 – Receiver Operating Characteristics for the classifier models used by the proposed system with the ANN based segmentation algorithm.
Table 5.12—Comparison between the classifiers performance attributes with the fused results using the majority-voting fusion method. The segmentation method used is the watershed based segmentation method and the execution time is the average execution time for 1010 images.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>% sensitivity</th>
<th>% specificity</th>
<th>% false positive rate</th>
<th>% total accuracy</th>
<th>Process (Segmentation, features extraction, and classification)</th>
<th>Execution Time/ image (Sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>86.09</td>
<td>50.23</td>
<td>49.77</td>
<td>68.16</td>
<td>0.284</td>
<td></td>
</tr>
<tr>
<td>ANN</td>
<td>82.17</td>
<td>44.29</td>
<td>55.71</td>
<td>63.23</td>
<td>0.282</td>
<td></td>
</tr>
<tr>
<td>KNN</td>
<td>63.08</td>
<td>67.58</td>
<td>32.42</td>
<td>65.33</td>
<td>0.278</td>
<td></td>
</tr>
<tr>
<td>D.Tree</td>
<td>33.25</td>
<td>89.50</td>
<td>10.5</td>
<td>61.37</td>
<td>0.279</td>
<td></td>
</tr>
<tr>
<td>AdaBoost</td>
<td>67.13</td>
<td>49.77</td>
<td>50.23</td>
<td>5845</td>
<td>0.356</td>
<td></td>
</tr>
<tr>
<td>Majority Voting fused model</td>
<td>71.68</td>
<td>64.38</td>
<td>35.62</td>
<td>68.03</td>
<td>0.0016*10^{-3}</td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.24—Receiver Operating Characteristics for the classifier models used by the proposed system with the watershed based segmentation algorithm.
In this research 10 ensembles of 3 classifier aggregates and one ensemble of five classifier aggregates are fused by the majority voting fusion method and evaluated by the DST for a trust factor and the one with the highest trust factor, smallest FPR, and highest accuracy is used as the system composite classifier model to classify a lymphocyte test image.

Table 5.13 shows the classification performance attributes results of the tested aggregates, in which the fusion of the KNN, the SVM and the decision classifier aggregate give the highest DST trust factor.

Table 5.13– The classifier ensembles with classification performance attributes and DST trust factor for the fused models.

<table>
<thead>
<tr>
<th>Classifier ensemble</th>
<th>Fused classifier model performance classification attributes</th>
<th>DST trust factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM, D.Tree, AdaBoost</td>
<td>% sensitivity: 82.68, % specificity: 87.21, % FPR: 12.79, % total accuracy: 84.95</td>
<td>0.8436</td>
</tr>
<tr>
<td>KNN, D.Tree, AdaBoost</td>
<td>% sensitivity: 82.05, % specificity: 88.58, % FPR: 11.42, % total accuracy: 85.32</td>
<td>0.8327</td>
</tr>
<tr>
<td>KNN, SVM, AdaBoost</td>
<td>% sensitivity: 85.84, % specificity: 85.84, % FPR: 14.16, % total accuracy: 85.84</td>
<td>0.8832</td>
</tr>
<tr>
<td>KNN, SVM, D.Tree</td>
<td>% sensitivity: 84.95, % specificity: 89.04, % FPR: 10.96, % total accuracy: 87.0</td>
<td>0.8416</td>
</tr>
<tr>
<td>ANN, SVM, D.Tree</td>
<td>% sensitivity: 84.07, % specificity: 89.04, % FPR: 10.96, % total accuracy: 86.56</td>
<td>0.8426</td>
</tr>
<tr>
<td>ANN, SVM, AdaBoost</td>
<td>% sensitivity: 86.09, % specificity: 84.93, % FPR: 15.07, % total accuracy: 85.51</td>
<td>0.8861</td>
</tr>
<tr>
<td>ANN, D.Tree, AdaBoost</td>
<td>% sensitivity: 83.69, % specificity: 84.01, % FPR: 15.99, % total accuracy: 83.85</td>
<td>0.8267</td>
</tr>
<tr>
<td>ANN, SVM, KNN, D.tree</td>
<td>% sensitivity: 83.44, % specificity: 89.5, % FPR: 10.5, % total accuracy: 86.47</td>
<td>0.8416</td>
</tr>
<tr>
<td>ANN, KNN, AdaBoost</td>
<td>% sensitivity: 84.83, % specificity: 84.93, % FPR: 15.07, % total accuracy: 84.88</td>
<td>0.8772</td>
</tr>
<tr>
<td>ANN, SVM, KNN</td>
<td>% sensitivity: 86.59, % specificity: 85.84, % FPR: 14.16, % total accuracy: 86.22</td>
<td>0.8871</td>
</tr>
<tr>
<td>ANN, SVM, KNN, D.Tree, AdaBoost</td>
<td>% sensitivity: 84.95, % specificity: 89.5, % FPR: 10.5, % total accuracy: 87.23</td>
<td>0.8416</td>
</tr>
</tbody>
</table>
5.5.8 Validation

The validation process is conducted in two phases. The first phase is conducted during the reinforcement learning where 799 pre-classified lymphocyte images are used to calculate the classification performance parameters as illustrated in Table 5.10. The second phase is a decision correlation between flow cytometry results and the proposed system classification decisions, in which a flow cytometry device is used to analyse blood samples from 11 patients (CLL cases).

Figure 5.25 shows the results of direct correlation between the pre-classified lymphocyte images and the system results, in which it shows a correlation R=0.66 for the two cell classes (CLL and normal). Table 5.14 shows the correlation for the individual cell classes, in which the R and P values for correlation and significance are illustrated and it shows that both individual cell class correlation has a significant results (P<0.05).

![Figure 5.25](image_url)

Figure 5.25 – Direct correlation between the manually pre-classified lymphocyte images and the classification of these images using the proposed system with the correlation R=0.66.
Table 5.14– Correlation between manual classification and the proposed system classification output showing the resulting correlation (R) and significance (P) values for the two lymphocyte cell class (CLL and Normal).

<table>
<thead>
<tr>
<th>Correlation</th>
<th>R value</th>
<th>P value (P&lt; 0.05= significant)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manually classified Normal lymphocyte VS Algorithmic Classified Normal lymphocyte</td>
<td>0.1932</td>
<td>0.0039</td>
</tr>
<tr>
<td>Manually classified CLL lymphocyte VS Algorithmic Classified CLL lymphocyte</td>
<td>0.0842</td>
<td>0.0178</td>
</tr>
</tbody>
</table>

The direct correlation between the number of the CLL cells identified by the flow cytometry and the number of CLL cells identified by the proposed system cannot be achieved, as it appears that the cells scanned by CellaVision™96 are not sampled the same way as the flow cytometry device. However, the flow cytometry data can be used to conduct a decision correlation in which other blood samples are acquired from the same 11 CLL cases and analyzed using the CellaVision™96. The proposed system is used to analyze the lymphocyte images with the hypothesis that if the percentage of CLL cells found is $\geq 70\%$ of the total lymphocyte cells, then there is a chance of CLL case, and if the percentage of CLL is less than 40\%, then the system advocates a normal case, and in between 40\% to 70\% a suggestion for re-examination of the blood smear for that case is supported by the system.

Table 5.15 shows the flow cytometry analysis results for the 11 CLL cases with the system suggestions in which the results show that the proposed system is capable of identifying the CLL cases in a way that matched with the flow cytometry results. In other words if the system identifies the majority – $\geq 70\%$ – of the lymphocytes as CLL there is a great possibility that the case under test is a positive CLL case.
Table 5.15 – Decision correlation between the proposed system results and the flow cytometry device results for 11 CLL cases.

<table>
<thead>
<tr>
<th>Case number</th>
<th>Slides number</th>
<th>Total CLL Cells</th>
<th>% Total CLL Cells found by the algorithm</th>
<th>Flow cytometry</th>
<th>CDSS system suggestion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ER0018 - 22</td>
<td>720</td>
<td>72.2</td>
<td>CLL</td>
<td>CLL</td>
</tr>
<tr>
<td>2</td>
<td>ER0023 - 27</td>
<td>1001</td>
<td>80.6</td>
<td>CLL</td>
<td>CLL</td>
</tr>
<tr>
<td>3</td>
<td>ER0028 - 32</td>
<td>574</td>
<td>84.3</td>
<td>CLL</td>
<td>CLL</td>
</tr>
<tr>
<td>4</td>
<td>ER0033 - 37</td>
<td>663</td>
<td>84.5</td>
<td>CLL</td>
<td>CLL</td>
</tr>
<tr>
<td>5</td>
<td>ER0048 - 52</td>
<td>623</td>
<td>73.8</td>
<td>CLL</td>
<td>CLL</td>
</tr>
<tr>
<td>6</td>
<td>ER0053 - 57</td>
<td>483</td>
<td>75.2</td>
<td>CLL</td>
<td>CLL</td>
</tr>
<tr>
<td>7</td>
<td>ER0058 - 62</td>
<td>546</td>
<td>78.7</td>
<td>CLL</td>
<td>CLL</td>
</tr>
<tr>
<td>8</td>
<td>ER0063 - 67</td>
<td>566</td>
<td>75.3</td>
<td>CLL</td>
<td>CLL</td>
</tr>
<tr>
<td>9</td>
<td>ER0068 - 69</td>
<td>45</td>
<td>31.1</td>
<td>CLL</td>
<td>Normal</td>
</tr>
<tr>
<td>10</td>
<td>ER0070</td>
<td>30</td>
<td>43.3</td>
<td>CLL</td>
<td>Re-examine</td>
</tr>
<tr>
<td>11</td>
<td>ER0071</td>
<td>84</td>
<td>79.8</td>
<td>CLL</td>
<td>CLL</td>
</tr>
</tbody>
</table>

5.6 System output

The proposed system main objective is to provide the hematopathologist with the percentage of the CLL and the normal lymphocyte cells present in the test images along with the suggestion of the system for the case/slide under examination. The report contains a graph, which represents a comparison between the chosen classifier models and the fused composite classifier model. The report contains a suggestion for the hematopathologist in which a CLL case is suggested when the system recognize the CLL cells ≥ 70% of the processed cells, or suggest a normal case if the system recognize the CLL cells ≤ 40% and the system suggests a re-examination of the case otherwise. Figure 5.26 shows a sample report of the test CLL case identified by the flow cytometry device.
5.7 Conclusion

The proposed system uses three segmentation algorithms (the SVM, the ANN, and the watershed), to segment the lymphocytes cell and further segmented into nucleus and cytoplasm. The SVM algorithm has a superior performance in terms of segmentation accuracy and segmentation processing time, and, thus, is selected by the proposed system to be the main segmentation algorithm.

Feature extraction deals with measuring the shape and texture properties of a given mask and provides the attributes (features) that the classifiers can use to learn how to separate the two classes, and therefore the distinctive feature will positively impact the
performance of the classifiers. FSAs are used to select the most distinctive features among the extracted ones.

The system uses five different classifiers of different structure and complexity. The proposed system uses an ensemble of these classifiers and fuses the results of the individual classifier models. The fusion is done using the majority voting fusion method and the ensemble performance is ranked using the DST calculated trust factor, in which the system uses the ensemble with the highest DST trust factor, smallest FPR, and highest CCR to classify a new test lymphocyte image. The system results are designed and validated using 1010 pre-classified lymphocytes images and 5535 lymphocyte images from a flow cytometry refereed 11 CLL cases.

The proposed system output is presented to the interpreting hematopathologist in the form of a report, in which it contains the results of every classifier along with the fusion results of the classifier ensemble. A table containing the percentage of CLL and normal cells found and a suggestion is also presented. The final decision is made by the hematopathologist.

Table 5.16 shows a comparison between the CDSSs proposed in studies [93][97] and the proposed CDSS in this research for CLL detection. The accuracy of the proposed system is 87.0% while the accuracy of the Leuko proposed in [93] is 72%. The proposed system uses less features (20) and classifiers (3) than the Leuko system (62 features and 5 classifiers [97]).

Considering running both systems on the same platform the proposed system will be faster. The accuracy of the proposed system is 3% less than that of the Leuko system.
with 5 SVMs classifier [97], and it was verified using more CLL images than the *Leuko* system. The ROC analysis in not available for studies [93][97].

The *Leuko* system is implemented in a concrete framework that does not allow any addition of a new algorithm. The cross-validation method used in [93][97] is the ‘Hold-out’ cross-validation, which tends to give optimistic results, whereas 10-fold cross validation method is used to verify the design of the proposed systems.

Table 5. 16 – Comparison of the proposed CDDS and other CDSS for CLL detection. (CVM= cross-validation method)

<table>
<thead>
<tr>
<th>study</th>
<th>Segmentation method</th>
<th>Number of used features/ FSA</th>
<th>CLL Accuracy/ FPR/ CVM</th>
<th>Number of classifier</th>
<th>Fusion method</th>
<th>Number of CLL images/ Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sabino et al [93]</td>
<td>Texture color segmentation</td>
<td>62</td>
<td>Accuracy =72% FPR= N.A Hold-out</td>
<td>1 (Bayesian)</td>
<td>N.A</td>
<td>Images =151 Resolution =720x480</td>
</tr>
<tr>
<td>Ushizima et al [97]</td>
<td>Texture color segmentation</td>
<td>62</td>
<td>Accuracy =90% FPR = N.A Hold-out</td>
<td>5 (SVMs)</td>
<td>Lorena etal[98]</td>
<td>Images =151 Resolution =720x480</td>
</tr>
<tr>
<td>The proposed CDSS</td>
<td>Watersher , SVM, and ANN</td>
<td>20</td>
<td>Accuracy= 87.0% FPR= 10.96% 10-fold</td>
<td>3 (KNN, SVM, and Decision tree)</td>
<td>Majority voting</td>
<td>Images =791 Resolution =363x360</td>
</tr>
</tbody>
</table>
Chapter 6: Conclusion, Limitations, and Future Works

In this chapter I conclude this thesis with the main achievements, results, and contributions; which will be followed by the factors that limit and restrain the results of this research. Finally, the future works of this study are illustrated which show the – big picture – of the extendibility and scalability of the proposed CDSS system, which can detect not only CLL but other types of blood cancer related to mutation in the WBCs and RBCs.

6.1 Conclusion

CLL is the most common type of blood cancer in Canadian adults. The relative 5-year survival rates of CLL in Canada is decreasing. The CLL is characterized by the progressive accumulation of functionally incompetent monoclonal lymphocytes in the bone marrow without responding to cell growth inhibitors. According to the clinical practice guideline of Alberta Health Care Services “LYHE – 007Version 2”; CLL/SLL is the most common adult leukaemia in the western world, accounting for approximately 7% of non-Hodgkin lymphomas [14] and the average age at diagnosis of CLL is 67 years; and it is rarely seen in children [14]. The clinical course and phenotypic presentation of CLL is highly diverse, and there are limited treatment options [14]. The current clinical practices delay treatment until a patient demonstrates either symptomatic or progressive disease, which do not necessarily correlate with the optimal treatment outcomes or long-term survival [2]. Despite being a common disease, there are few studies examining the automated detection of CLL from digitized peripheral blood films.
The scope of this research is the analysis of lymphocyte microscopic blood images to detect CLL cells. The system consists of lymphocyte segmentation, feature extraction/selection, and multiple classifiers system, and the system output is presented in a report format. The design and performance of the system has been validated using 6,345 microscopic blood images, and these images are grouped as follows: 1010 manually pre-classified lymphocyte images and 5,335 images from CLL cases refereed by flow cytometry device. The proposed system can segment the lymphocyte cells with high accuracy, and label the cell as CLL or normal. A report is presented to the hematopathologist containing the percentage of the found CLL and normal lymphocyte cells in a graphical representation along with a suggested decision.

The manual scanning process is tedious and time consuming; as a hematopathologist has to study around 100 to 150 microscopic views of Giemsa-stained thin blood smear images per slide to detect a CLL case. The proposed system provides a tool to support the decision of the hematopathologist in a fast and efficient way.

This research has demonstrated a method for segmenting lymphocyte cells (normal and CLL) using the watershed algorithm and Otsu’s optimal thresholding technique. The effect of over and under segmentation of the watershed algorithm is significantly reduced by removing 1% of the local minima before applying the watershed algorithm. The experimental results show that the proposed method is able to yield 98.41% maximum accuracy for nucleus segmentation and 99.99% maximum accuracy for cell segmentation. With simple pixel to pixel subtraction, the cytoplasm mask can be accurately driven by subtracting the nucleus mask from the whole cell mask. The maximum accuracy for cytoplasm segmentation is 99.96%.
In this research a method for lymphocyte color cell segmentation using the SVM algorithm and ANN is discussed. The most important aspect of the SVM and the ANN as a machine learning algorithm is their capabilities to overcome the occlusion problem when lymphocytes are tightly bound to the surrounding RBCs. Over and under-segmentation problems are thereby significantly reduced. For segmentation accuracy measurement, 140 images are used, and 12 of these images are used for training of the SVM and the ANN algorithms. The SVM algorithm has a superior segmentation performance, in which it obtains 97.0% ±0.5 average accuracy for nucleus segmentation, and 97.62% ±0.77 for cell segmentation. The cytoplasm region can be extracted by 92.08% ±9.24 average accuracy with simple mask subtraction.

There are two types of the FSAs which are filter type and wrapper type. The main objective of the FSAs is to remove the redundant features that may negatively impact the classification process, and to reduce dimensionality of the classifier input data, which helps decreasing the processing time of the classification process and increasing the CCR.

MCS is a technique used to enhance the results of the classification process, in which an ensemble of classifiers is trained using part or all of the training dataset in a parallel, serial, or hybrid fashion and used to classify a test image, and then the results of the ensemble are fused using a fusion method which can be a majority-voting, minimum, maximum, and product. The idea of enhancing the classification results is based on the fact that every classifier may make different mistakes while classifying a test lymphocyte image. The idea of classifiers ensemble concentrates on the aggregate performance of the classifier models rather than the individual performance of every classifier model. The
DST is used to calculate a trust factor for every possible ensemble of the classifier models used by the MCS in a 3 and 5 classifiers group. The used classifier aggregate composed of the SVM, the KNN, and the decision tree classifier, which have the highest DST trust factor, which is 84.16 % and the following classification performance attributes: 89.04% specificity, 84.95% sensitivity, 10.96% FPR, and 87% over-all accuracy.

The system output is represented in a report format containing the classifier performance and a suggestion is presented to the hematopathologist, however the final decision is made by the interpreting hematopathologist.

The SA of the proposed system is based on the data-centric architecture as it perfectly fits the nature of the CDSS, where many algorithms can access the data and modify it then this modified data can be used by another algorithm independently. The data-centric architecture provides a flexible architecture to data size growth and reusability of the system. The adaptive framework accompanied by the tailored wrapper design pattern provides a flexible generic tool to hook a new algorithm into the system, in which the wrapper pulls the variables of the new algorithm which may be written in plain MATLAB® script, OO code, MEX, DLL, and EXE and let the system user to do the matching between the input and the output.

The proposed system can be used as a cheap and fast pre-screening tool to delineate the CLL cells in a blood smear slide, which will significantly reduce the reliance on the flow cytometry devices as they are costly, non-reproducible results, and complicated screening tool.
6.2 Limitations

The lymphocyte images used in this study are acquired from the commercial hematopathology equipment CellaVision™96 which impose some image quality attributes such as image resolution, objective settings and specimen illumination, and thus the results of this research are dependent on these settings. The CellaVision™96 acquires the images with the cell at the center of the image, which facilitate the localization of the cell in the segmented mask, however if this is not true for any other equipment settings a search technique for the cells must be used otherwise some large stain spots located near the center of the image are going to be recognized as cells. One possible search technique is the Hough transform [101] which may be used to search for circular shapes, and may work well if the cell is almost round in shape, however some cells may take a complicated non-circular shape which makes it difficult to localize, and, thus, a more superior localization technique must be used to precisely locate the cell in the segmented mask.

There are three algorithms used to segment the lymphocyte images which are the watershed, the SVM and the ANN, in which more images with different staining and illumination conditions must be examined to test the performance of the segmentation algorithms using a wide range of input images. In order to enhance the nucleus segmentation accuracy; multiple thresholds or local thresholding must be used to overcome the Otsu’s assumption of bimodal histogram and uniform illumination. Moreover, controlled watershed markers must be used to prevent the washing away of the lymphocyte cell into the background, resulting from the adhesion of the lymphocyte cell with many RBCs.
The proposed algorithm based on the SVM and the ANN have some limitations, when two or more lymphocytes are touching, in which case the algorithm identifies it as one entity. More inference rules must be utilized to overcome this problem. However the overall performance of the algorithm is quite promising for accurate lymphocyte color cell segmentation. Examples of touching lymphocytes and the corresponding resulting masks are shown in Figure 6.1.

The classification methods used in this research are based on the non-parametric supervised classifier models, and thus the results are limited to the training dataset training algorithms, the reinforcement learning process, and the optimization method used by every classifier model.

The fusion method is limited to maximum-majority voting method with the DST calculated trust factor of the classifier ensemble.

The system can use the tailored wrapper design pattern to interface to a new added algorithm in DLL, EXE, MEX, and only MATLAB® 2011b as a script language, other script languages like PHP and JAVA are not tested.

The software quality attributes addressed in this research are only the re-usability, flexibility, and extendibility of the system. Other attributes such as security and reliability measures are not discussed in this study.
Figure 6.1– Touching lymphocytes and corresponding output (a) Original images, (b) Extracted cell mask, (c) Extracted nucleus mask, (d) Extracted cytoplasm mask.

### 6.3 The Future works – The Big Picture –

The future works of this research are to acquire many more CLL images, which describe the CLL disease in different stages. This will add capabilities to the system to model the mutation process and assess the treatment plan and to categorize a newly found CLL cell to belong to which stage in the disease cycle. The following future works will be the validation of the whole system operation by different clinical settings and different pathologists; and this will be achieved by installing the proposed system on a cloud server; and use it from different clinical sites, which requires preparing the system to be accessed by different users. The security issue as a non-functional requirement now becomes a concern; and difference in image quality and specimen staining must be considered.

The big picture of the proposed system is to make it available and to be used by different hematopathologists. This big picture also involves the hosting of different algorithms that may be used to identify other types of blood cancer and for conducting research.

The big picture of the system which I call– *EzPatho* – is illustrated in Figure 6.2.
Figure 6.2– The big picture of the “EzPatho” generic CDSS.
REFERENCES


[8] www.cellavision.com


[76] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994.


